Faculdade de Engenharia da Universidade do Porto

Development of an Optimized Omnidirectional Walk Engine for Humanoid Robots

Nima Shafii

Programa Doutoral em Engenharia Informática

Supervisors:

Luís Paulo Reis, EEUM / LIACC - University of Porto

Nuno Lau, University of Aveiro / IEETA

24 de Janeiro de 2015

Abstract

Wheeled robot locomotion is not adapted to many human areas, e.g. areas littered by many obstacles, while humanoid robots can be adapted to human environments since they can attain desirable posture and avoid obstacles of different shapes.

Due to the huge controller design space as well as inherently nonlinear dynamics of walking, controlling the balance of humanoid walking is a very challenging task. In addition to the walking balance issue, in order to use humanoid walking in daily life tasks, humanoids should also be capable of walking in an optimized manner, e.g. energy efficient and high speed.

There is a large number of approaches to generate humanoid walking, which are mainly divided into model-based and model-free approaches. However, there is still not a clear methodology to address the issue of generating an optimized omnidirectional walking in the sense of creating a fast and energy efficient walk.

This thesis is intended to develop biped locomotion approaches in order to generate an optimized omnidirectional walking. Model-free approaches based on Truncated Fourier Series (TFS) and model-based approaches based on Zero Moment Point (ZMP) are studied to generate optimized walking. The TFS based approach is improved to generate three-dimensional walking motions using all legs degrees of freedom. This improvement enables the robot to perform turn-in-place and to walk faster when compared with the previously presented TFS based approaches.

ZMP-based approaches are also presented which can generate omnidirectional walking using the cart-table model. An analytical approach is presented to solve the differential equations of the cart-table model allowing the robot to change the double support period during the walk.

The cart-table cannot model hip height movement during walking. However, in this thesis, a ZMP-based approach is presented which can model an omnidirectional walking with variable hip height. The hip height movement is generated using Fourier series and Central Pattern Generators (CPGs) that had been used in model-free walking approaches. Gait optimization approaches are applied to the proposed ZMP-based walk engine with variable hip. This enables the robot to walk faster and also to walk with higher energy efficiency in comparison with the walking generated using the regular cart-table model.

The proposed model-based and model-free walk approaches, were implemented and tested on both simulated and real humanoid robots. The results showed that by combining and using the techniques developed in model-free and model-based approaches, the robot could perform an optimized omnidirectional walking, concerning energy efficiency and with higher speed.

The proposed optimized walk engine approach was also used in our humanoid soccer robot team that competes in RoboCup international competitions. Our results are encouraging given the fact that the robot performs very well, being able to walk fast and stable in any direction in comparison with the best RoboCup 3D soccer simulation teams, which use the same simulator. Our RoboCup simulation team could achieve first place in three consecutive RoboCup European competitions in 2012, 2013 and 2014, and third place in 2013 International 3D Soccer Simulation league.

Resumo

A movimentação de robôs com recurso a rodas não é adequada a muitas áreas preparadas para humanos, tais como áreas obstruídas por obstáculos. No entanto, robôs humanóides podem ser adaptados a ambientes humanos, dado que conseguem uma postura desejável e evitar diferentes tipos de obstáculos.

Devido ao enorme espaço de estados do controlador bem como à dinâmica não-linear inerente à marcha, controlar o equilíbrio durante a marcha de um robô humanóide é uma tarefa muito desafiadora. Além da questão do equilíbrio, por forma a usar marcha em tarefas diárias, os humanóides devem ser capazes de caminhar de forma otimizada, ou seja, eficiente a nível energético e com uma velocidade elevada.

Existe um grande número de abordagens para criar marcha humanóide propostas na literatura, e que estão essencialmente divididas em "baseadas em modelo" e "sem modelo". No entanto, não há ainda uma metodologia clara para abordar a questão da criação de uma marcha omnidirecional otimizada que permita um deslocamento rápido e eficiente a nível energético.

Esta tese teve o objetivo de desenvolver abordagens de locomoção bípede, a fim de criar uma marcha omnidirecional otimizada. Abordagens "sem modelo" baseadas em Séries de Fourier Truncadas (TFS) e abordagens "baseadas em modelo" baseadas no "Ponto de Momento Zero" (ZMP) foram estudadas para criar uma marcha otimizada. A abordagem baseada em TFS foi melhorada de modo a gerar movimentos de marcha tridimensionais usando todos os graus de liberdade das pernas do robô. Esta melhoria permite que o robô possa executar rotações no próprio local e andar mais rápido em comparação com as abordagens baseadas em TFSs anteriormente apresentados.

São apresentadas também abordagens baseadas em ZMP que criam uma marcha omnidirecional utilizando o modelo de "carrinho na mesa". Uma abordagem analítica é apresentada para resolver as equações diferenciais do modelo "carrinho na mesa" permitindo que o robô altere o período de suporte duplo durante a caminhada.

O modelo de "carrinho na mesa" não consegue modelar alterações na altura da anca durante a caminhada. No entanto, nesta tese, uma abordagem baseada em ZMP é apresentada que pode modelar uma marcha omnidirecional com uma altura variável da anca. O movimento vertical da

anca é gerado usando séries de Fourier e “Geradores de Padrão Centrais” (CPG) que foram usados em abordagens de marcha “sem modelo”. São aplicadas abordagens de otimização ao modelo de marcha baseada em ZMP, com altura de anca variável, proposto. Isso permite que o robô se desloque mais rápido e também com uma maior eficiência energética em comparação com a marcha gerada usando o modelo tradicional de “carrinho na mesa”.

As abordagens “baseadas em modelo” e “sem modelo” propostas, foram implementadas e testadas tanto em robôs humanóides reais como em simulações. Os resultados mostraram que ao combinar e usar as técnicas desenvolvidas em abordagens “baseadas em modelo” e “sem modelo”, o robô pode executar uma marcha omnidireccional otimizada, relativamente à eficiência energética, e com uma velocidade mais elevada.

Os modelos de marcha otimizados foram usados na nossa equipa de futebol robótico humanóide que participam em competições internacionais RoboCup. Os resultados obtidos são encorajadores dado que os robôs da equipa conseguem um muito bom desempenho no que diz respeito a andar rápido e de modo estável em qualquer direção, em comparação com as melhores equipas de simulação de futebol RoboCup 3D, que utilizam o mesmo simulador. A nossa equipa de simulação RoboCup conseguiu também o primeiro lugar em três competições consecutivas Europeias de RoboCup, em 2012, 2013 e 2014, e um terceiro lugar em 2013 na liga Internacional de Simulação de Futebol 3D.

Acknowledgments

This work was funded by the Portuguese Foundation for Science and Technology (FCT) under the grant SFRH / BD / 66597 / 2009, gratefully acknowledged. I offer my special thanks to my supervisor Prof. Dr. Luis Paulo Reis for his guidance and generous support throughout this work. Also, I would like to express my sincere gratitude to my supervisor Prof. Dr. Nuno Lau for his inspiration and excellent guidance.

I would also thank the Institute of Electronics and Telematics Engineering of Aveiro (IEETA) for its supporting of my work in the last year of my Ph.D. in the concept of the grant PEst-OE/EEI/UI0127/2014. Moreover, I acknowledge the (current and former) members of both the Artificial Intelligence and Computer Science Laboratory (LIACC) and IEETA, in particular Rui Ferreira, Abbas Abdolmaleki, and Edgar Domingues, whose scientific discussions were certainly a plus.

Concerning the non-academic side of my life, my respectful thanks are to my loving parents Abdoulreza Shafii and Vajiheh Rezagholinejad for their constant encouragement and support in all stages of my studies.

Last but certainly not least, I would like to thank my beloved wife Shirin. She made a fundamental contribution to the development of this work. I managed to finish this thesis mainly because of her excellent support.

"Your hand can seize today, but not tomorrow; and thoughts of our tomorrow are nothing but desire. Don't waste this breath, since the rest of your life won't last forever."

Omar Khayyam

Table of Contents

List of Figures

List of Tables

List of Abbreviations

2D	Two-dimensions
3D	Three-dimensions
CMA-ES	Covariance Matrix Adaptation Evolutionary Strategy
CMP	Centroidal Moment Pivot
CoM	Center Of Mass
CMP	Centroidal Moment Pivot
CPG	Central Pattern Generator
CZMP	Center of ZMP region
DARPA	Defense Advanced Research Projects Agency
DEEC	Departamento de Engenharia Electrotécnica e de Computadores
DRC	DARPA Robotics Challenge
DSP	Double Support Phase
EC	Evolutionary Computation
EEUM	Escola de Engenharia da Universidade do Minho
FEUP	Faculdade de Engenharia da Universidade do Porto
GA	Genetic Algorithm
GCoM	Ground projection of the Center of Mass
HC	Hill Climbing
IEEE	Institute of Electrical and Electronics Engineers
IEETA	Institute of Electronics and Telematics Engineering of Aveiro
IP	Inverted Pendulum
IPM	Inverted Pendulum Model
LIACC	Artificial Intelligence and Computer Science Laboratory
TFS	Truncated Fourier Series
ZMP	Zero Moment Point
LIPM	Linear Inverted Pendulum Model
MAE	Mean Absolute Error
ODE	Open Dynamics Engine
OpenHRP	Open Architecture Humanoid Robotics Platform

PD	Proportional Derivative
PFS	Partial Fourier Series
PID	Proportional Integral Derivative
PSO	Particle swarm Optimization
SA	Simulated Annealing
TFS	Truncated Fourier Series
TS	Tabu Search
VRML	Virtual Reality Modeling Language
ZMP	Zero Moment Point

Chapter 1

Introduction

In robotics, a humanoid robot is a robot made to be similar to a human both in appearance and behaviour. In general, humanoid robots' body structure resembles human's body parts, including head, torso, legs and arms, which allow the robot to perform biped locomotion. Bipedal locomotion is a form of locomotion where a humanoid robot moves by means of its two legs. This movement includes any type of omnidirectional walking or running.

In humanoid robotics, the number of joint actuators indicates the number of Degrees of Freedom (DOF). They often have 10 to 14 DOF for the legs, 3 at each hip, 1 DOF for each knee and 2 or 3 for the ankle [1][2][3]. Like humans, humanoid's body moves in three planes including transverse (axial), frontal (coronal) and sagittal plane. Figure 1.1 shows a schematic view of a humanoid robot. DOFs 1, 3, and 4 move in Sagittal plane and DOFs 2 and 5 move in Frontal plane and DOF 6 moves in the transverse plane.

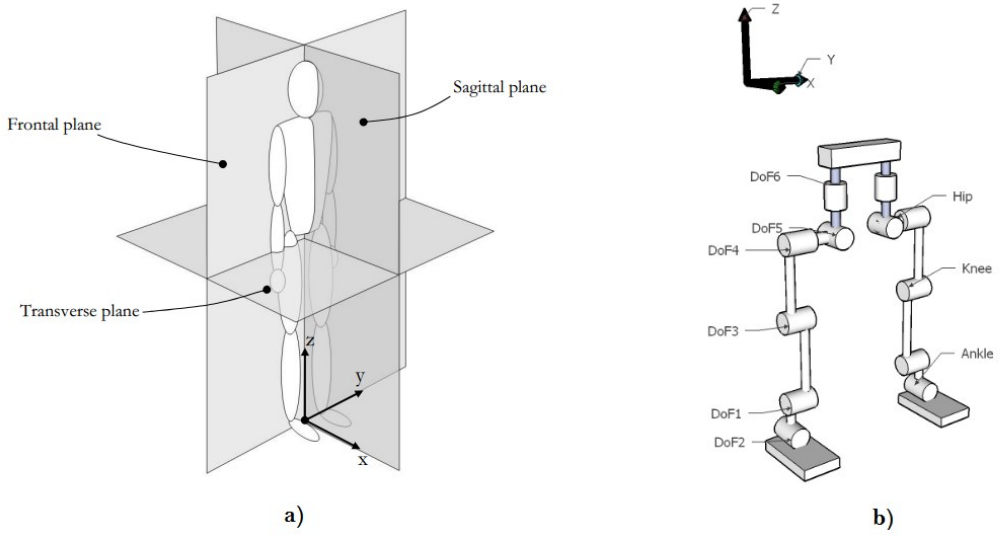

Figure 1.1: a) Human body in different Planes [4], b) Schematic view of a humanoid robot

A walking has different phases, with respect to support mode it consists of single leg support and dual leg support phase, and with respect to leg motion it consists of stance and swing phase. In Figure 1.2, the distinct phases of human walking are shown.

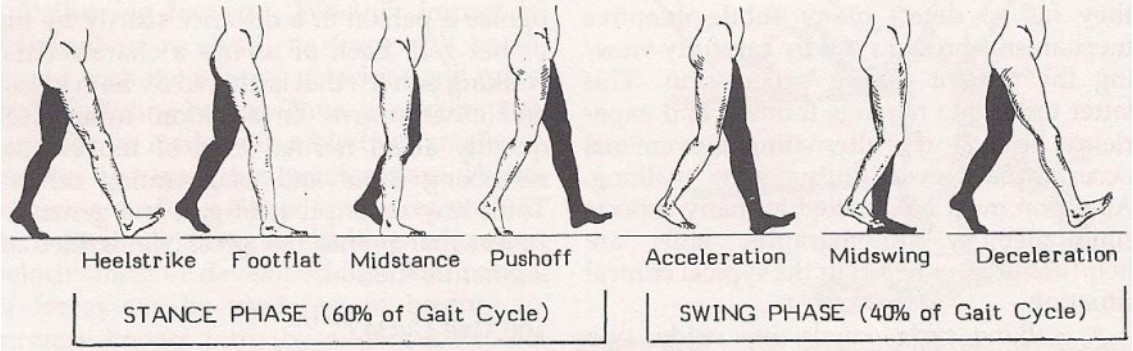

Figure 1.2: Leg cycle phases [5]

The leg swing phase corresponds with single support phase but stance phase is not the same as dual support phase, stance phase will include double support and also partial single support. In Figure 1.2, where the heel strike posture is shown, the human is in double support phase.

In this section, properties of a humanoid gait were given, in the rest of the chapter we will address the motivations, main objectives and contributions of this thesis.

1.1 Motivation

Forty years after the advent of the first autonomous humanoid robot that could perform three-dimensional biped walking [6], because of recent achievements in biped walking approaches and the availability of smaller and cheaper humanoid robots, the research on biped walking is renewing itself more strongly and innovatively than before. Humanoid robots are designed in a human-like fashion. They have a large number of joints that makes them attractive to researchers.

While wheeled robots locomotion is not adapted to many human environments such as, stairs and areas containing many obstacles, humanoid robots are able to function and do their tasks easier than wheeled robot in areas designed for people. Humanoid robots have, in general, a larger number of degrees of freedom (DoFs) than wheeled robots, which enables them to walk, climb stairs and attain proper posture for avoiding obstacles in an easier manner.

In addition, according to the fact that biped locomotion is similar to human movement, this similarity causes people to interact easier with humanoid robots than with other types of robots. From an energy efficiency viewpoint, it has been shown that humans walking on two legs only used one-quarter of the energy than chimpanzees who knuckle-walked on four legs did [7]. This

fact supports the hypothesis that walking on two legs, or bipedal walking, evolved because of its energy efficiency [7]. Humans, like other natural systems, could not survive the competition and natural selection proposed by Darwin's theory of evolution without being capable of energy-efficient locomotion.

Even though biped locomotion has many advantages in the view of the aforementioned aspects, it still has not been used in real common tasks, such as industrial applications and military activities. In the next section, we will talk about the reasons behind why biped locomotion approaches are still under discussion and for what reasons biped robots cannot walk as human can.

1.2 Biped Locomotion Challenges

Humanoid robots must be able to adjust their walk speed and direction to perform their tasks, and their walk must keep its balance and adapt itself considering different terrain and surfaces. Humanoid robots also have different shapes of body and properties. Thus, the approach for generating a stable walk for humanoid robots must deal with all of these issues.

Humans can walk and run quite well, and very few of them had any control courses or teaching on how to perform stable walk. However, humanoid robots cannot run or walk as stable as humans can. "A walk may be considered as stable if the only contact between the biped and the floor is performed with the soles of the foot or feet, i.e. no other extremity of the biped is in contact with the floor" [4]. The main question concerning this matter may be seen as why the control of biped walking and running is still challenging and has not been solved. Humans can learn to walk by applying natural algorithms that are developed through practice. In contrast, humanoid robots, usually, use complex control algorithms designed by researchers or engineers to perform their bipedal locomotion without being able to apply the same set of natural algorithms that enable humans to learn to walk in a very stable manner.

There are different types of gait patterns that can be applied in humanoid walking or running. This redundancy is created because humanoid robots contain many joints DoFs. Although this redundancy provides freedom to choose different stable gaits in walking, controlling all of DoFs simultaneously in order to generate a stable walk is still a very challenging problem. This difficulty is caused by several distinct factors. First, the dimension of the design space of the stability dynamic controller is usually quite large. Second, the nature of the walking dynamics is nonlinear. Typically, in nonlinear control systems, classical control techniques are applied on a robot, based on the idea that feedback can be used to override the dynamics of the robot [8]. However, for fully actuated systems, with known dynamics, it is possible to use feedback

linearization methods as an effective design tool for changing a nonlinear control problem into a linear control problem [9].

From the viewpoint of mechanical control theory, legged locomotion can sometimes be considered as an underactuated system. Underactuated means that the number of available control actuators, or, more specifically, the number of independent generalized forces is fewer than the number of DoFs [10]. In other words, underactuated control systems are those in which the control input cannot accelerate the state of the robot in arbitrary directions. Consequently, unlike fully actuated systems, under actuated systems cannot be commanded to follow arbitrary trajectories.

Biped locomotion is considered as an underactuated system in several situations. First, when a robot's leg contacts the ground at only a point, there are not actuators on the point of the foot that is in contact with the ground. In this case, if too much torque is applied at the ankle, the foot will likely turn about one of its edges [11]. Therefore, this limits the acceleration of DoFs of the robot and consequently it reduces the ability to keep the robot from falling over and use the control policy to keep its balance [12]. Second, while a humanoid robot is running, at regular points during its cycle both feet are off the ground for some short time, then none of those legs' actuators are able to change the trajectory of the center of mass, therefore the robot will also be underactuated in this case.

Since designing a controller for underactuated systems is very difficult, until now many approaches on biped locomotion have considered the biped robot a fully actuated system. Results achieved by these kinds of approaches generate gait patterns for humanoid walking in a very conservative way. One of the conservative ways used by them is the generation of the walk at a low speed with both feet kept parallel to the ground.

Another disadvantage of using these approaches is that their generated walk is not energy efficient. In order to change a walking machine to a fully actuated system, it is needed to use a very high gain feedback in all legs' actuators to cancel out the walking dynamics caused by the gravity and walk acceleration. However, the dynamics caused by gravity may even help the walk execution, e.g. passive dynamic walkers can walk down slopes without any actuation and just by using gravity forces. Passive dynamic walker is the dominant biped locomotion example, in which the legged robot is considered as a fully underactuated system [13].

In summary, the main difficulties related with bipedal robot locomotion control can be summarized by the very large controller design space, interaction with an unknown environment, and the underactuated nonlinear control system [14]. Due to these facts, it is difficult to control the balance of the humanoid robot while it performs an omnidirectional walk.

1.3 Objectives

Nowadays, some commercial biped robots have been presented, but bipedal walking approaches still require more development to enable them to be implemented and work in a hard and complex real environment like, for example, a robot soccer domain.

Energy efficiency, higher locomotion speed, versatility and robustness are some of the main issues in creating omnidirectional walking, which should be studied in more depth in order to allow a widespread application of legged locomotion in autonomous systems. Therefore, this thesis intends to achieve the following main goal:

- To develop new approaches for creating fast, robust and energy efficient omnidirectional bipedal locomotion with a flexible methodology that can be implemented both on simulated and real humanoid soccer robots.

The main hypothesis addressed in this thesis is presented below:

> *"The development of model-based and/or model-free approaches and their use together with gait optimization techniques can produce fast, stable and energy-efficient omnidirectional humanoid robots' walking"*

In order to verify this statement and to achieve the presented goal, the following intermediate objectives are also defined:

- Review relevant works in the area of biped locomotion and gait optimization;
- Develop an optimized model-free gait generator able to create fast and robust biped locomotion using gait learning approaches;
- Bridge the gap between results of model-free approaches in simulation and real humanoid robots;
- Develop an omnidirectional walk engine based on model-based approaches using models that describe the walking dynamics;
- Improve the analytical approaches for solving the dynamical equation of motions of a balanced walking;
- Improve model-based approaches to create energy-efficient and fast walk using gait optimization methods;
- Evaluate the outcome of the model-based approaches and model-free approaches in realistic humanoid soccer scenarios.

According to our goals the main related scientific areas of the thesis are the following: Humanoid Robotics, Biped Locomotion, Nonlinear Dynamics, Central Pattern Generator, Gait Optimization, and Robot Learning.

1.4 Hardware Platform

This study is intended to present an approach to biped walking. The main goal is to present a method that can be applied on humanoid locomotion on a soccer domain. Agility of movement is one of the key factors which can improve the humanoid ability to compete in a demanding domain such as the soccer domain.

RoboCup is one of the most fascinating educational initiatives which are focused on the research on soccer robotics. RoboCup competitions include many different leagues. One of its major leagues is soccer humanoid robot league, where humanoid robots try to compete against each other. Biped locomotion is one of the keys of success in this competition. The goal of this study is limited to any type of humanoid robot locomotion, which can be used on the soccer domain. Therefore, the approach has to be able to enable the robots to move in any direction on a flat train.

The Humanoid robot platform used in this study is the NAO robot [2] with the specification of the RoboCup edition. NAO RoboCup edition is a kid size humanoid robot that has 21 degrees of freedom. It is 58 cm height and weighs 4.3 kg. The specification of the NAO robot that is used in this thesis can be found in Table 1.1.

Table 1.1: NAO robot specification

Albaderan NAO RoboCup edition	
Height	58 centimetres (23 in)
Weight	4.3 kilograms (9.5 lb)
Autonomy	60 minutes (active use), 90 minutes (normal use)
Degrees of freedom	21
CPU	Intel Atom @ 1.6GHz
Built-in OS	Linux
Compatible OS	Windows, Mac OS, Linux
Programming languages	C++, Python, Java, MATLAB
Vision	Two HD 1280x960 cameras
Connectivity	Ethernet, Wi-Fi

In this thesis, biped locomotion approaches first will be implemented in a simulation environment using Simspark [15] to simulate the NAO robot. The description of this simulator and the intuition behind using it will be presented in section 4.3.

1.5 Contributions and Achievements

Our contributions in this thesis are the following:

- Develop a novel gait generator model based on model-free approaches. The joints' angular trajectories are modeled by using a Truncated Fourier Series (TFS) without considering the physical model of the robot. Gait optimization techniques are used to optimize the model parameters for walking in a single direction and performing turn-in-place. The TFS approach had been presented before but only for a planar robot. We extended the TSF model to use all DoFs, which led us to generate 3D walking motion. Results for a fast forward walk are published in [16] and for a fast turn-in-place are published in [17].
- Develop an approach to bridge the gap between the walking results achieved in simulation and reality. There is a gap between the results of simulation and reality, especially when a model-free approach is used. For both forward walk and turn-in-place, the results achieved in simulation are modified in order to be used on a real humanoid robot [18] [19].
- Develop an omnidirectional walk engine based on model-based approaches, which mainly consist of a foot planner, a Zero Moment Point (ZMP) generator, Center of Mass (CoM) generator, and an active balance loop. A cart-table model is used to generate the CoM reference trajectory from the predefined ZMP trajectory. An active balance method is presented which tries to keep the robot's trunk upright when faced with environmental disturbances. Although the cart-table model was used to generate the omnidirectional walking, the new design of foot planner and active trunk balance control is proposed. This approach was tested on NAO robots in a soccer field [20].
- Develop analytical approaches for solving the differential equations of the cart-table model. This approach is based on the Fourier approximation of the ZMP trajectories, which lead us to generate the CoM trajectory faster and more accurately than numerical approaches, for both omnidirectional and single direction walk scenarios. In single direction walking, the approach had been previously proposed for straight and curve walking. We extended this approach to generate diagonal walking [21]. In omnidirectional walking, an analytical approach is presented, in which, by using a new time segmentation approach, parameterization of the double support phase is allowed [22]. Other researchers had previously proposed a general form of Fourier series based approach, but they did not parameterize different double support periods.

- Improve the efficiency of the walk generated by model-based approaches regarding the speed and energy efficiency. The hip height movement is used by a robot to achieve a faster or more energy-efficient walk. For the first time, the hip height movement is modeled in a formal way, and its parameters are learned by gait learning methods. The inverted pendulum model and a numerical approach are used to control the balance of the generated walk. A comparison of the results of the proposed gait model with those obtained using fixed hip height shows that fixed height walks are slower and less energy-efficient than variable height walks [23] [24].
- Develop an approach to generalize the learned results, by using a programmable Central Pattern Generator (CPG). The programmable Central Pattern Generator (CPGs) is used, in order to modulate trajectories smoothly for generalizing the results achieved by different learning scenarios. In one study, CPG is used to obtain not only fast forward walk but also a fast side walk [25]. In another study, the CPG is used to generate energy-efficient walking while the robot changes its walking speed [26][27].

Besides these contributions, the work presented in this thesis is also used in our soccer humanoid robot team. It allowed us to use an agile and robust omnidirectional walk in our participation in soccer robot competitions (RoboCup). Our team could achieve several awards in Worldwide and European RoboCup competitions.

1.6 Thesis Structure

This thesis is organized in eight chapters, in which the first one is composed by this introduction. The remainder of this thesis is organized as follows:

Chapter 2 discusses the well-established stability criteria that have been used for analysing walking stability.

Chapter 3 introduces the main biped locomotion approaches that had been used before by other researchers. A literature review of the biped locomotion approaches, which are used in our methodology, is given in detail.

Chapter 4 provides a comprehensive review of the literature about algorithms and techniques used in gait learning and gait optimization. A review about the current well-known humanoid simulators is also presented in this chapter.

Chapter 5 presents our contribution to the field of model-free biped locomotion. The methodological approach for modeling the walking trajectories and the results of gait optimization for both forward walk and turn-in-place are given in this chapter.

Chapter 6 presents our omnidirectional walk engine based on a model-based approach using the cart-table model. In addition, our novel analytical approach for solving the equations of motion of cart-table model is also described in this chapter. Results are given for both single-direction and omnidirectional walks.

Chapter 7 presents our contribution to the generation of an efficient walk by using the inverted pendulum model and gait learning methods. The use of Central Pattern Generator (CPG) to generalize the learning result is also discussed in this chapter. The results are given for both fast walking and energy-efficient walking scenarios.

Chapter 8 summarizes the main conclusions of this thesis. In addition, reflections about the limitations and some future perspectives are discussed.

Chapter 2

Locomotion Stability Measurements

As mentioned in Chapter 1, keeping the stability of a humanoid robot during its walk is one of the most important issues in biped locomotion studies. It is useful to be able to measure the quality of a gait in terms of stability in a quantitative way. The stability criterion can be used for evaluating the balance of a gait. It can also be used in designing a feedback controller to keep the balance of walk. Many stability measurement techniques use foot support area in their measurement approaches. It is also called the support polygon. This area is shaped by a convex hull, which is determined by the contact points of the robot's feet on the ground. Walking stability criteria often specify points located on the ground. Many stability measurements techniques analyse the distance between these points and the dimensions of the support foot area. In this chapter, well-established stability measurement techniques and well-known walking stability criteria will be reviewed.

2.1 Ground Projection of Center of Mass

The Ground projection of Center of Mass (GCoM) is defined as the projection of the center of mass along the gravity axis onto the ground plane. The GCoM of a humanoid robot can be calculated by using the equation (2.1) and (2.2)

$$m = \sum_{i=1}^{n} m_i \tag{2.1}$$

$$OG = \frac{1}{m}\sum_{i=1}^{n} OG_i . m_i \tag{2.2}$$

Here, m_i is the mass of the i_{th} body parts, and OG_i denotes the position of GCoM of the i_{th} segment which is expressed in coordinate system O and located on the ground plane. In biped

locomotion studies, a biped walk is called a static walk, when the GCoM always falls within the convex hull of the foot support area [28]. In this condition, the walk is statically stable and the posture is stated to be balanced [29]. This concept cannot be extended to dynamically stable walking. Indeed, the GCoM may leave the support area for pushing the robot to move forward during dynamic walking.

When robot is in stationary state, the exit of the GCoM from the support polygon may have a direct bearing on an uncompensated moment on the foot, which may cause the foot to rotate about a point on the polygon boundary. Although the position of the GCoM is adequate to determine the occurrence of foot rotation in a stationary robot, it is not so for a robot in motion [11]. Other stability indicators can be used for analyzing the stability of dynamic walking. The Zero Moment point is one the most well-known dynamic walking stability indicators, which will be discussed in the next section.

2.2 Zero Moment Point

Zero Momentum Point (ZMP) was presented in January 1968 by Miomir Vukobratović at The third all-union congress of theoretical and applied mechanics in Moscow [30]. The position of the ZMP is widely used for dynamic stability measurement in biped locomotion. Zero Moment Point (ZMP) was termed by the point on the ground at which the influence of all forces acting on the mechanism can be replaced by one single force [31], and the tangential component of the moment generated by this force becomes zero [32]. In order to make the mechanism analysis easier, let us assume a humanoid in the single support phase with the whole foot being on the ground, and considering that the foot on the ground is at rest without sliding.

Figure 2.1 shows a schematic view of the forces acting on foot in single support phase. Then, we can neglect the part of the mechanism above the ankle of support foot and represent its influence by the Force F_A and moment M_A. The foot itself also has a gravity force which acts at its center of mass denoted by G. The foot also experiences the ground reaction at point *P*, whose action keeps the whole mechanism in equilibrium. The total ground reaction consists of three components of the force $R\ (R_x,R_y,R_z)$ and moment $M\ (M_x,M_y,M_z)$.

For ensuring the dynamic equilibrium of the system, the support foot must fully rest on the ground. Thus, all forces and moments act on the foot must be balanced. The static equilibrium equations for the supporting foot are stated in (2.3) and (2.4).

$$R + F_A + m_s g = 0 \tag{2.3}$$

$$\overrightarrow{OP} \times \vec{R} + \overrightarrow{OG} \times m_s g + M_A + M_Z + \overrightarrow{OA} \times F_A = 0 \tag{2.4}$$

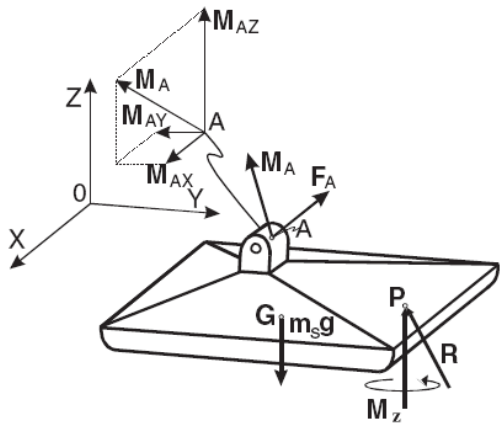

Figure 2.1: Biped mechanism and forces acting on the support foot [31]

By considering the proposed assumption that the foot floor contact is without sliding, the horizontal reaction force (R_x,R_y) represents the friction force that is balancing the horizontal component of the force F_A. Thus, M_z, the vertical component of the moment generated by frictional force, will also compensate the moments produced by horizontal force components of F_A. Since the foot also rests on the ground, reaction force R_z balances vertical forces. It remains to consider the balancing of the horizontal component of the foot load moment, which can be compensated by changing position of the reaction force R. Therefore, the computed Zero Moment Point is defined as that point on the ground at which the net moment of the inertial forces and the gravity forces has no component along the horizontal axes [31]. In reality, the point P cannot exist outside the support polygon or support foot area. As in that case, the reaction force R cannot act on the system, therefore the computed ZMP outside of the support polygon physically is not possible, and, the computed ZMP is considered as a fictitious ZMP (FZMP) [31].

ZMP is affected by the referred mass and inertia and kinematics of the robot's torso and links. Zero Momentum Point can be calculated by equations (2.5) and (2.6)[33], (Px, Py) are the position of the ZMP in the Cartesian domain on the Floor:

$$p_x = \frac{\sum_{i=1}^{n} m_i(\ddot{z}_i + g)x_i - \sum_{i=1}^{n} m_i \ddot{x}_i z_i - \sum_{i=1}^{n} I_{iy}\ddot{\Omega}_{iy}}{\sum_{i=1}^{n}(\ddot{z}_i + g)m_i} \tag{2.5}$$

$$p_y = \frac{\sum_{i=1}^{n} m_i(\ddot{z}_i + g)y_i - \sum_{i=1}^{n} m_i \ddot{y}_i z_i - \sum_{i=1}^{n} I_{ix}\ddot{\Omega}_{ix}}{\sum_{i=1}^{n}(\ddot{z}_i + g)m_i} \tag{2.6}$$

In this Equation, x_i, y_i, and z_i indicates the position of the center of mass of link i in Cartesian coordinate system with z-axis pointing up. g is the gravitational constant and m_i is the mass of link i. I_{ix} and Ω_{ix} are the centroid moment of inertia and angular displacement about the X axis, respectively. Angular displacement of a body is the angle in radians which a point or

line is rotated in a specified sense (about) a specified axis [34]. I_{iy} and Ω_{iy} are also assumed as the related parameters about the *Y* axis.

For a given set of walking trajectories, if the ZMP trajectory keeps firmly inside the area covered by the foot of the support leg or the polygon containing the support legs (boundaries of stability region), the given biped locomotion will be physically feasible and the robot will not fall over during the motion. When ZMP is outside of the support polygon, the ground reaction force acting point *P* is located actually on the edge of the support polygon and the robot is rotated about the support polygon edge.

Unbalanced moment causes the locomotion mechanism to start to rotate about the foot edge and the mechanism can collapse. This unbalanced moment intensity depends on the distance from the support polygon edge to the computed position of ZMP, i.e. to the FZMP position. Therefore, the ZMP must be kept within the stability region [31].

ZMP criteria cannot distinguish between a marginally stable equilibrium and an unstable situation since both states are located at the boundary of the support polygon [11]. In other words, it can be used as criteria for stability, but not for instability. The Foot Rotation Indicator, which will be explained in section 2.4 tries to overcome this problem.

The most common use of the ZMP is in trajectory planning. In section 3.2, we will overview the method of ZMP based approaches on biped locomotion task. The general idea of the ZMP is also used in on-line reactive stability control [35].

2.3 Center of Pressure

The Center of Pressure (CoP) is defined as a point on the foot or ground surface where the net ground reaction force actually acts. It is possible to show that the CoP is the point where the resultant moment generated by the inertial and gravity forces is tangential to the surface and the final result of moment around that point in the horizontal direction is zero [11]. Therefore, ZMP and CoP are indicating the same point when the robot is balanced. Regardless of the state of stability of the robot, the CoP may never leave the support polygon [11]. The Center of Pressure can be determined by using equation (2.7) .

$$OP = \frac{\sum_i q_i . F_{ni}}{\sum_i F_{ni}} \tag{2.7}$$

Where *OP* is the vector from the origin of the coordinate system *O* to the Center of Pressure position, q_i is the vector to the point where force F_{ni} acts in the direction of the surface normal (*Z* direction). In another word, F_{ni} is the component of the ground reaction force, which is perpendicular to the surface of contact. Figure 2.2 represents the ground reaction force, where

the normal forces and the frictional tangential forces are shown in left and center part of the figure. The CoP is the point *(P)* where the normal forces R_n acts. At the CoP, the tangential forces are interpreted by a resultant force R_t, which acts tangentially over a surface. The ground reaction force is equal to the summation of R_n and R_t.

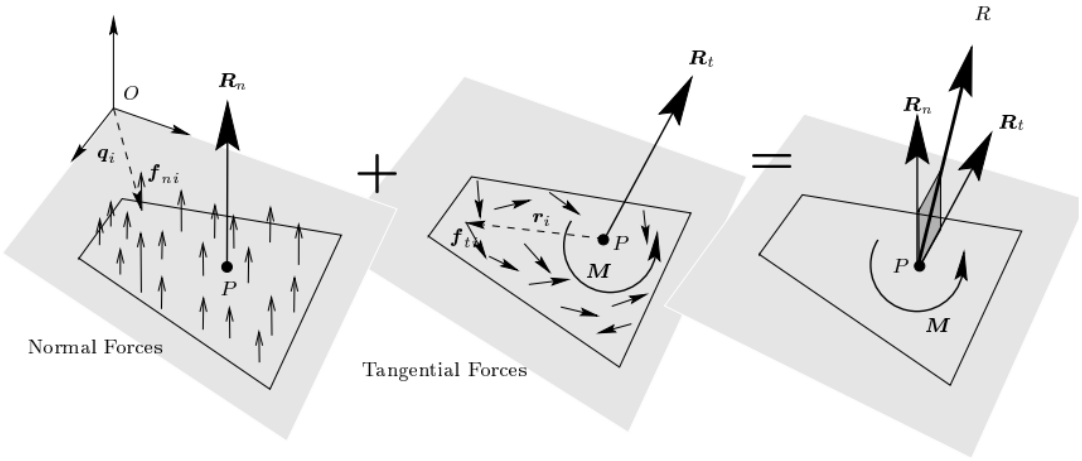

Figure 2.2: Center of Pressure, tangential forces and ground-reaction force [11]

2.4 Foot Rotation Indicator

The Foot Rotation Indicator (FRI) can be an indicator to measure postural instability. It indicates the severity of torque making the foot to rotate, and it calculates a point on the foot or ground surface where the net ground reaction force would have to act to keep the foot stationary [11]. FRI point can be inside or outside the support polygon, the farther away this point is from the support boundary the larger is the unbalanced moment, and the greater is the instability. In fact, the distance of the FRI point from the support polygon is an indication of the severity of this unbalanced torque[11]. To ensure no foot rotation, the FRI point must remain within the support polygon, regardless of the GCoM position. Although GCoM position can just be applied to a static walking method, the FRI can also be applied to dynamic walking approaches to ensure the stability of walking. In equation (2.8) and (2.9) the foot rotation indicator position (F_x,F_y) defined in coordinate system *O* is calculated.

$$OF_x = \frac{m_1 OG_{1y} g + \sum_{i=2}^{n} m_i OG_{iy}(\ddot{x}_{iz} + g) + \sum_{i=2}^{n} m_i OG_{iz}\ddot{x}_{iy} + \sum_{i=2}^{n} \dot{H_{Gix}}}{m_1 g + \sum_{i=2}^{n} m_i(\ddot{x}_{iz} + g)} \quad (2.8)$$

$$OF_y = \frac{m_1 OG_{1x} g + \sum_{i=2}^{n} m_i OG_{ix}(\ddot{x}_{iz} + g) + \sum_{i=2}^{n} m_i OG_{iz}\ddot{x}_{ix} + \sum_{i=2}^{n} \dot{H_{Giy}}}{m_1 g + \sum_{i=2}^{n} m_i(\ddot{x}_{iz} + g)} \quad (2.9)$$

In the above equations, OF_x and OF_y indicate the position of the FRI on the ground. Here, the C*oM* of a humanoid robot is created of *n* links denoted by *i* and *i=1* indicates the foot. m_i is the

mass of segment *i*, $\ddot{x}_{iz}$ is the linear acceleration of segment *i* and $\dot{H_{OG_i}}$ is the angular momentum about the center of mass OG_i defined in the coordinate system O.

As described in the previous sections, when a biped is dynamically balanced, ZMP and CoP are located on the same point. CoP represents the point on the foot or ground where the net ground reaction force actually acts and may never leave the convex hull of the foot support. The FRI point is defined as the resultant moment of force/torque impressed on the foot. It is normal to the surface and can leave the convex hull of the foot support.

In Figure 2.3, the definition of each criterion is shown in the sketch of a rigid-body biped robot. The CoP, GCoM and the FRI point are denoted by *P, A* and *F*, respectively. Where *OP* is the vector to the position of *ZMP* and *CoP* defined in coordinate system *O, G1* is the center of mass of the foot.

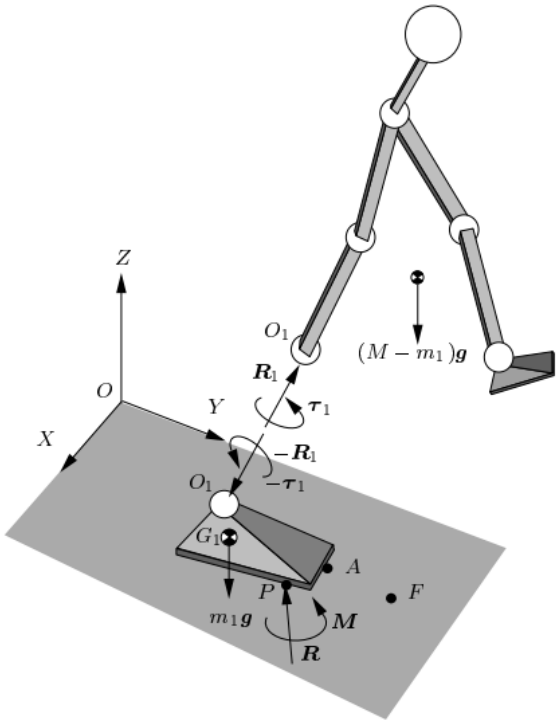

Figure 2.3: The sketch of a 3-D extended rigid-body biped robot [11]

2.5 Centroidal Moment Pivot

One of the problems with the FRI is that it is only defined during the single support phase. The Centroidal Moment Pivot (CMP) overcomes this problem. The CMP is based on the finding that the body's angular momentum about its center of mass remains very small throughout a stable gait [36]. Therefore, the amount of angular momentum can be used to determine robot instability.

Given that the ground is planar, the CMP is defined as the point where a line parallel to the ground reaction force, passing through the center of mass of the robot, intersects with the ground. As such, it provides information on body rotation. The CMP can be written in terms of the center of mass and the ground reaction force which is expressed in Equations (2.10) and (2.11).

$$C_x = CoM_x - \frac{F_{GRx}}{F_{GRz}} CoM_z \tag{2.10}$$

$$C_y = CoM_y - \frac{F_{GRy}}{F_{GRz}} CoM_z \tag{2.11}$$

Where *C* denotes the CMP Position, CoM is the Center of Mass and $F_{GR}$is the ground reaction force. When the moment about the center of mass is zero, the CMP coincides with the CoP. In this situation the robot is stable, thus, the CMP and ZMP are also located on the same point. The distance between the CMP and CoP is equal to the magnitude of the horizontal component of moment about the CoM, divided by the normal component of the ground reaction force. Therefore, this distance can be used as a measure of instability of walking.

2.6 Poincaré maps

The dynamical system of a biped robot can always be expressed by Newton's and Euler's equations as differential equations; therefore limit cycle studies can be useful. The most well-known tool for analysing limits cycles of the continuous periodic dynamical systems is the Poincaré map. A Poincaré map of an *N* dimensional state space defines a mapping from the *N-1* dimensional state at a Poincaré state onto itself. In addition, a Poincaré section is a hyper-plane in the state space which is transverse to the flow of the dynamical system. In stability analysis the periodic motion is defined as stable when the eigenvalues of the linearized Poincaré map (the Jacobian matrix) are within the unit cycle. In this case, the system will converge to a fixed point on the Poincare section. However, to be able to derive the stability properties, the Poincaré map and Jacobian matrix of the full dynamical system must be known or approximated.

For any given bipedal periodic gait, a Poincaré section can be defined as a certain point in the gait cycle. Given any state variable of the biped robot, a state variable of the controller or a measured quantity such as center of pressure, the assumption can be made that stability is directly related to the variance of the distribution of state variables intersecting the Poincaré section during locomotion. The more unstable the gait is, the larger the variance in certain state variables will be.

Poincaré analysis is often used when the exact model of the robot and the environment are known and usually expressed as equations of motion [37], or to derive the stability of only the controller. However, it is not suitable for analysing the system in the case of interaction with the environment. Although the Poincaré map and the Jacobian matrix method is often not applicable in practical situations where there are unknowns about the environment and the model of the robot, the concept can still be applied to measure stability.

Recently many works are presented to find the stability margin and region-of-attraction analysis for nonlinear hybrid limit cycle like biped walking by approximating a Poincaré map of the dynamics of biped walking. By using machine learning approaches, robots learn to find policies to get back to the limit cycle again against any environmental perturbation [38]. Finding the way to optimize the region-of-attraction for nonlinear hybrid limit cycles such as biped walking system is still a challenging task in optimal control studies [39]. Nowadays, optimal control and analysing the Poincaré maps is the most common approach for controlling underactuated systems. Although it can be categorized as a feed-forward control policy, recently researchers tried to improve these kinds of approaches by adding some feedback control laws [40].

2.7 Summary

All presented methods can be used to evaluate stability while walking generated by any controller. Although, the GCoM can only be used for analyzing the stability of a static walking. The FRI, ZMP, CMP and CoP can evaluate the stability of both dynamic and static walking. The FRI, ZMP and CMP need to know accurate kinematic and dynamic state for each body part of the robot. Many humanoid robots do not have accurate sensors in order to estimate this information. In simulation, it is possible estimate kinematic and dynamic state based on the information given by the physical simulator. Therefore, only in simulation and offline mode, it is possible to use these measurements of stability to evaluate a certain gait controller.

On the other hand, the CoP can be calculated when the robot has force sensors at its feet. As it was mentioned, the ZMP and CoP are located at the same point, when the robot is stable. Therefore, by using only force sensors at feet, the position of the ZMP can be estimated in real robot walking in an online mode. In this case, ZMP can be used both in active stabilization approaches in both real robot tasks and in simulation to evaluate accurately stability of a certain gait.

The boundary of the region where the ZMP of a stable walk should be inside it has been well defined. In contrast, in FRI and CMP criteria, the boundary of the stability region has not been clear. Since the ZMP dynamics equations give the relation between CoM movements and ZMP

movements and the ZMP of a stable walk can be predesigned, the ZMP equations can be used in a feed-forward control system in order to generate the CoM of a stable walk. The ZMP equations are used in many approaches to generate an stable walk. In Chapter 3 we will overview the ZMP based approaches. In this thesis, we use the ZMP as the stability criteria for designing our balance controller that will be discussed in Chapter 6 and Chapter 7.

Chapter 3

Biped Locomotion Approaches

In this chapter, biped locomotion approaches will be introduced as the background of this thesis. Over the years, various methods for handling bipedal locomotion have been proposed. In the first section, we will overview biped locomotion approaches briefly and will determine which of them will be used in this thesis. In the rest of the sections, we will discuss fully the ones that are decided to be used in the methodological part of this thesis.

3.1 Overview

Biped locomotion approaches can be classified by using their three main characteristics that are given in the following categories.

Model-based vs Model-free approaches: In model-based approaches, first, the kinematics and dynamics of a biped robot is modeled; then, a control algorithm is implemented on the model in order to produce stable walk. In model-free approaches, without building a kinematics and dynamics model of a robot, a stable walking motion is generated by mapping between sensors and actuators. In this thesis, both types of the approaches will be used.

Passive vs Active walkers: In passive walker, all robots' joints are passive. It means that there is no actuation on the joints. This type of the walking approach assumes the walk machine is a fully under-actuated system. The goal of researches in this area is to study how to use the benefit of the natural dynamic of the walk, in order to create most energy efficient walk.

The passive walking approach was first presented by McGeer [41]. He built a 2D passive walker that could walk down on the slight slope only by using of gravitational forces. This design is known to be the simplest walk model namely the "compass gait model". Collins et al. designed the first 3D passive walker which used swinging arms to counteract the lateral

instability [13]. Up to now, interesting results have been achieved by using this type of approaches, more complex prototypes showed that passive designs are able to produce remarkably human-like gait without any active power [42] [43], but they all lead to several disadvantages. In passive walking, the gait cannot be controlled directly, meaning the legged robot cannot control walking direction and speed, in addition, only by using the gravitational forces it is not possible to walk on level ground, in this case, actuation is needed. As the result, the passive dynamic approach has not been used in any applicable and commercialize humanoid robot yet.

In active walker, the walking machine is assumed a fully actuated system, the robot joints' actuation is active, and all joints are controlled by motors. In this case, the walk patterns are generated first, specifying all joints trajectories, then the joints' actuators are used to follow these trajectories. Walk pattern can be changed online, therefore the robot can control the direction of walk. Active walking approaches are still the most common approaches used in the humanoid robot locomotion engine [1][2][3]. In this thesis, we will focus on designing an active walker.

Static vs Dynamic Gaits:

In static walk approaches, only the position of the CoM is considered to build the walk pattern generator. It assumes that the walk is in balance when the Ground projection of Center of Mass (GCoM) is always kept in the support polygon defined by the support foot/feet. However, when a robot walks fast with significant acceleration, the GCoM leave the support polygon. In static gait, the whole-body dynamics of the humanoid robot do not help the stability during the walking motion, because the CoM must always stay close to the middle of the support polygon. In this case, the CoM trajectory is almost the same as the ZMP trajectory. Graf et al. [44] presented a static walk approach which is implemented on the NAO humanoid robot, they also used feedback from sensors to improve stability.

In dynamic walk approaches, the position and acceleration of the CoM are considered, and the walk is generated by controlling the dynamic of the walking motion. A dynamic gait is more similar to a human gait. The dynamic walk approaches can generate a faster and more human-like walk than the walk that is generated by a static walk approach. In this thesis, we will focus on the approaches that can generate dynamic walking. The ZMP criterion, presented in section 2.2, is one of the most popular stability indicators, which is used to analyze the stability of a dynamic gait.

The walking pattern generators based on ZMP are one of the most popular dynamic walking approaches [45] [46] [3] [47]. They belong to model-based, active walker group. The ZMP based approaches will be used in this thesis and will be explained in the section 3.2 in detail.

In model-free approaches, we will also use the approaches the can generate a dynamic and active walk. Truncated Fourier Series (TFS) [48], bio-inspired Central Pattern Generators (CPG) [49] are among the well-known model-free approaches that are able to generate dynamic gaits. Sections 3.3 and 3.4 will discuss the mentioned model-free approaches.

3.2 ZMP based Approaches

Many popular approaches used in bipedal locomotion are based on the Zero Moment Point (ZMP) stability indicator. This section will overview ZMP based approaches. This type of approaches will be used in our methodological part of this thesis, in which a walk engine will be designed that can generate active dynamic walking. The ZMP criterion has been extensively used as the stability measurement in the literature [33]. For a given set of walk trajectories, if the ZMP trajectory keeps steadily inside the area covered by the foot of the support leg or the convex hull containing the support feet, this walk will be physically stable and the robot will not fall. When the ZMP reaches the edge of the polygon, the robot loses its balance.

The ZMP is a very significant theory in motion planning for biped robots. It is a well-defined methodology for insuring the stability of robot during walk. Since using ZMP stability indicator decreases the probability of a robot to fall during its walk, it is well-suited for expensive Robots, for instance, Asimo (Honda Robot) controls the target ZMP, which maintains the robot's stability [45][50]. The Asimo robot can keep its balance even when the floor is uneven or when pushed by small external forces [45]. Figure 3.1 shows the evolution of the Asimo robot over time.

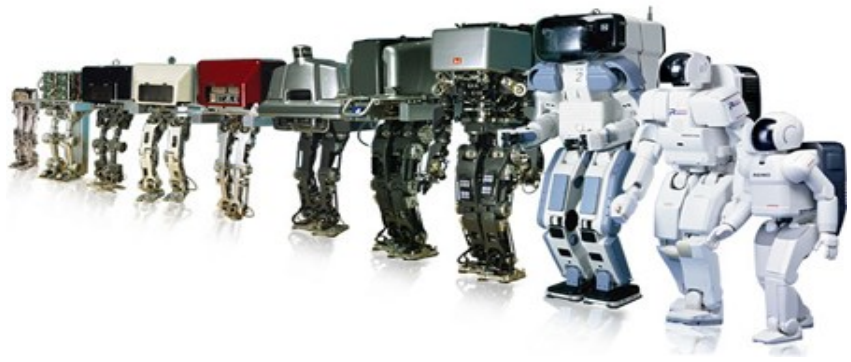

Figure 3.1: Asimo robots (Honda Company) are well-known humanoid robots

Having the ability to generate an omnidirectional walk that consists of straight, curved, side and diagonal walk, and being able to change the direction of the walk are requirements in humanoid robots to do their task in a dynamic environment. A daily-life environment usually is a dynamic

environment so that humanoid robots need to avoid movable obstacles, in this regard, humanoid robots need to be able to perform omnidirectional walk to avoid these obstacles. ZMP based approaches have been used to create omnidirectional walk [51][32][46]. Since soccer humanoid robots play in a dynamic environment soccer field, ZMP based methods are implemented on them to generate omnidirectional walk [52][46][53][54]. It has also been reported that the ZMP based approaches can generate running and jumping motions [55][56].

ZMP can be used in biped locomotion approaches in two different main categories; in the first type of approaches, ZMP does not generate walking trajectories directly but it can indicate whether generated walking trajectories will keep the balance of robot or not. Thus, the main challenge still exists which is to find proper gait trajectories that create a stable walk and respect the ZMP constraint [57] [58] [59]. In this type of ZMP-based approaches, The ZMP location is commonly used as a feedback mechanism to achieve stable gait via either foot pressure sensors or online modeling [60].

In the second type of approaches, biped walking is obtained through the modeling movement of ZMP and CoM. Biped walking trajectories can be derived from predefined ZMP trajectory by computing a CoM trajectory that produces this ZMP trajectory. This CoM trajectory can be approximated by using the dynamic models of simplified physical systems. Concentrated-mass models are among the popular models used for this approximation. The Concentrated-mass models simplify the whole body dynamics to a CoM motion by concentrating the robot mass into a single point. The motivation behind using these models is to reduce the model complexity and computational time.

It is difficult to derive whole body dynamics and apply them to obtain accurate walking patterns, and it is highly challenging to control those dynamics to create a stable walk. Since, humanoid robots' computation power is normally limited and whole body dynamics control is hard to carry out in real-time. Therefore, researchers turned to simplified dynamic models, which are suitable for onboard real-time planning and control.

The Cart-Table model [61] and simple inverted pendulum model [62] are the popular concentrated-mass models that have been used to generate CoM trajectory by approximating the bipedal robot dynamics. In this section, both Cart-Table model and Inverted pendulum model are explained in detail, since we use these two models in our research, they are presented in Chapter 6 and Chapter 7.

3.2.1 Cart-Table Model

In this section, Cart-Table model will be explained in order to respond the question how is it possible to use the ZMP in biped locomotion. In the single support phase, humanoid walking

motion can be represented by the Cart-Table model [61]. Cart-table model has some assumptions and simplifications. First, it assumes that all masses are concentrated on the cart. Second, it assumes that the support leg does not have any mass. Although these assumptions seem to be far from reality, modern walking robots usually have heavy trunks with electronic circuits and batteries inside. Therefore, the effect of leg mass is relatively small. Figure 3.2 shows how robot dynamics is modeled by a Cart-Table model.

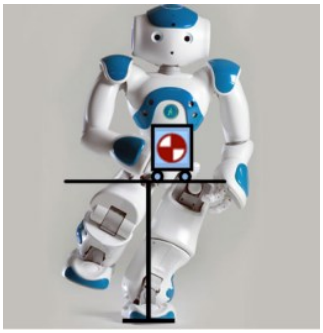

Figure 3.2: The Cart-Table model used on a NAO robot

Two sets of Cart-Table models are used to model 3D walking. One is for movements in sagittal plane; another is for movements in coronal plane. The schematic view of a Cart-Table model, which is used in sagittal plain (XZ plane), is shown in Figure 3.3.

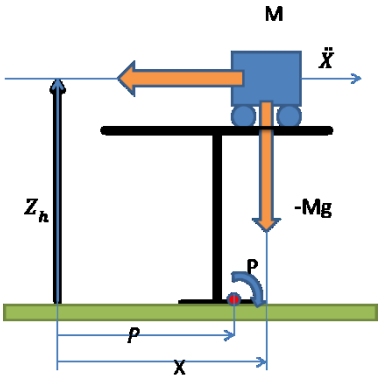

Figure 3.3: Schematic view of Cart-Table model

The position of the Center of Mass (CoM) M is x and Z_h defined in the coordinate system O. Gravity g and cart acceleration $\ddot{x}$ create a moment T_p around the Center of Pressure (CoP) position P. Equation (3.1) provides the calculation of the produced moment or torque around P.

$$T_p = Mg(x - P) - M\ddot{x}z_h \tag{3.1}$$

According to section 2.2 we know that when the robot is dynamically balanced, ZMP and CoP are identical, therefore the amount of the moment in the CoP Point must be zero, $T_P=0$. By assuming the left hand side of equation (3.1) to be zero, the equation (3.2) provides the position of the ZMP on X axis.

$$P_x = x - \frac{Z_h}{g}\ddot{x} \tag{3.2}$$

By using the same deduction way, the position ZMP in *Y* axis is given in equation (3.3).

$$P_y = y - \frac{Z_h}{g}\ddot{y} \tag{3.3}$$

A 3D Linear Inverted Pendulum Model (LIPM) for generating humanoid walk has been first introduced in [63]. It has been shown that LIPM has a close similarity with Cart-Table model and has the same equations that are given in (3.2) and (3.3) [61]. LIPM is constrained to move along a plane, meaning it cannot have any horizontal movements that result in linear state space equations.

Figure 3.4 shows the schematic view of LIPM. LIPM assumes that the ZMP is in the origin, *O*, that corresponds to the robot's ankle and the ankle torque is Zero [64]. In contrast, in Cart-Table model, the ZMP can move in the foot that is modeled by the table contacts with the ground, and the torque of ankle in not needed to be zero.

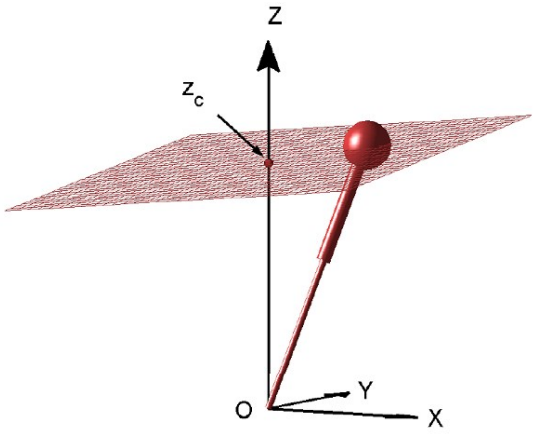

Figure 3.4: A linear inverted pendulum model [63]

As it was discussed in Chapter 2, The ZMP position of a stable walk should be located inside the support polygon, therefore the ZMP trajectory of a stable walk can be obtained from its feet placements. Then, the walk is generated by following the CoM position trajectory that is calculated by solving the equations (3.2) and (3.3). In this case, these two equations are the differential equations where the ZMP trajectory (*P*) is predefined and the position an acceleration of CoM trajectory ((x,y) and $(\ddot{x}, \ddot{y})$) are the unknowns parameters that should to be calculated. There is not a straightforward way to compute CoM from ZMP by solving these differential equations. The approaches presented previously on how to tackle this issue are organized into two major groups; numerical and analytical approaches.

As a numerical approach, Kagami et al. proposed a gait generation technique by descretizing the differential equation of the relationship between the ZMP trajectory and CoM

trajectory [65]. Kajita et al. presented an approach to find the proper CoM trajectory, based on the preview control of the ZMP reference, which makes the robot able to walk in any direction [61]. This optimal method obtained a good result using forthcoming values of the desired ZMP [66] and it is a dominant numerical approach that has been used widely in literature to generate biped walking [53][67]. In Chapter 6, we will use ZMP preview control approach to generate CoM trajectory in the implementation of our proposed omnidirectional walk engine [20], a detailed explanation of ZMP preview control approach is given in section 6.3.2.

Several analytical approaches have also been well established and investigated till now. Kurszume et al. calculated gaits based on the analytical solution of the differential equations [68]. Even though, the exact solution of the Cart-Table differential equations can be calculated theoretically, applying this solution to generate the CoM reference trajectory is not straightforward. Because the solution consists of unbounded *cosh* functions, and the calculated CoM trajectory is very sensitive to the time step variation of the walk. Takanishi et al. presented a method for generating the trunk motion by transforming ZMP reference trajectories into the Fourier series [69]. Erbatur et al. investigated a similar approach to calculate the CoM trajectories, in which the solution of the differential equations of Cart-Table are approximated based on the Fourier representation of the ZMP equation, this approach could create straight and curved walk [70] [71].

All mentioned analytical approaches are categorized as offline methods, which can generate CoM reference trajectory before a robot actually moves. They are not able to connect the CoM trajectories of different walk directions smoothly and in online manner. Even the approach that was presented by Erbatur et al. [70] was not able to generate walk in any direction. Recently we improved it, in order to be able to generate diagonal walk [21]. A detailed explanation of this improvement is given in section 6.2.1.

Although a humanoid robot is able to walk in a single direction using offline methods, but generating the walk in any direction and having the ability to change them online, improves the ability of a robot to avoid obstacles, which is necessary to do in daily-life tasks. Nishiwaki et al. presented an analytical approach based on a method for modifying the gait pattern on-line by connecting the current reference trajectories to the newly calculated one [72].

Herada et al. has improved similar techniques, in which the ZMP reference trajectory of a walk is represented by a spline function and an analytical solution of the differential equation was calculated [32]. Since these approaches segment the ZMP trajectories, they are called time segmentation based approaches. Park et al. have used these techniques with the Fourier series for representing the ZMP trajectory [73]. The method realizes the smooth connection of the trajectories by simultaneously calculating the ZMP and CoM trajectories, and it can also

approximate many different shapes of the ZMP trajectory, because of the nature of the Fourier series. Although the general form of Fourier based approach is explained in [73], it is not specifically explained how to parameterize different double support periods. We have presented a new time segmentation design to parameterize different double support periods [22]. We will explain this approach in details in section 6.3.3.

Numerical approaches seem not to be completely suitable for generating CoM reference trajectories used in the soccer humanoid robot walking, due to their inherent time complexity. It becomes difficult to calculate the gait within one sample time, especially for changing the direction of the walk in a real time task. In contrast, analytical approaches do not have this issue.

The common way that researchers apply the Cart-Table model on a biped locomotion is given as follows; first foot placements are calculated by a foot planner algorithm. the desired ZMP trajectory is derived based on the foot placements. Then, the CoM trajectory is calculated by the Cart-Table model. After that, positional trajectories of the swing foot can be computed via a cubic Spline generalization [74] or Bézier curves [75]. These methods guarantee a smooth trajectory with respect to linear and rotational velocities [74]. Then, inverse kinematics is used to find the angular trajectories of each joint based on the position trajectory of the feet and the calculated CoM position. Finally, the servomotors and proportional-integral-derivative controller (PID controllers) are used to control the movement of joints' motors to actively follow the robot joints' angular trajectories.

According to the fact that the Cart-Table model has some simplification and it is not very accurate, some recent work was done to get online feedback to robust robot walking. It uses actual ZMP position calculated by foot pressure sensors during the walk, and tries to correct the movements to aid the robot to follow planned ZMP trajectories [46]. The major drawback of using Cart-Table model and LIMP is its simplification that the CoM has a constant height and cannot move vertically. In this case, the supporting knee must be bent during swing phase, as opposed to a natural way of human walking.

The bent knee causes that heavy extension torque is needed to support body weight. Figure 3.5 shows a gait with constant CoM height which needs high knee torque. The high torque and large joint motion leads to more than doubling of energy expenditure compared with normal walking [76]. Carey et al. have confirmed there is a higher metabolic cost for walking in this way [77]. In the next section, we will explain the inverted pendulum model that can produce a varied height walk and remedy the disadvantages caused by walk with constant height.

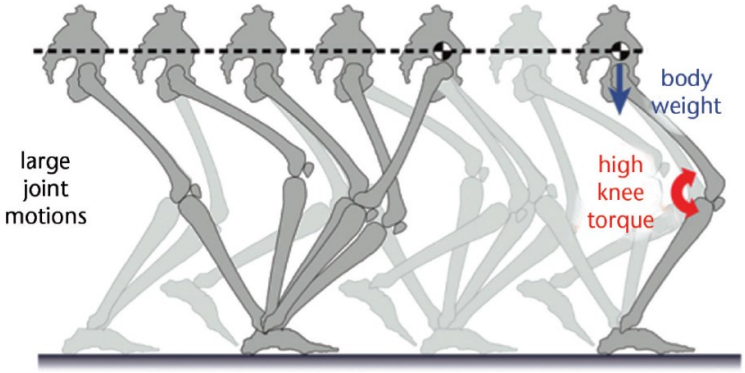

Figure 3.5: High knee torque is needed to walk with constant height [77]

3.2.2 Inverted Pendulum Model

Biped walk can be represented as the problem of balancing an inverted pendulum model since in the single supported phase human walking can be represented as an inverted pendulum [78]. The major drawback of using the Cart-Table model is that it considers the robot's CoM height fixed during its walk. This fixed height restriction is not true for many biped locomotion types, e.g. running, walking. Inverted pendulum model prevents this issue; in this section, we explain inverted pendulum model.

The simple inverted pendulum model has fixed length mass-less rod. In the biped locomotion studies, this model is extensively used to study the biped walking [79] [78][80]. A more general form of inverted pendulum model used in the literature is a model where the connection between the pivot point and the CoM is assumed to be a mass-less telescopic rod. In its modelling of a humanoid walk, this model has some assumptions and simplifications. First, like other concentrated-mass models, it assumes all masses are concentrated on the trunk of the robot. Second, it assumes that the support leg is mass-less. Figure 3.6 shows the NAO humanoid robot and the inverted pendulum model with telescopic rod.

Figure 3.6: Frontal view of the NAO robot and the inverted pendulum model with telescopic rod

Figure 3.7 shows a schematic view of the inverted pendulum model in *XZ* plane. Two sets of inverted pendulum are needed to model a 3D walking. One is for movements in the coronal plane; another is for movements in the sagittal plane.

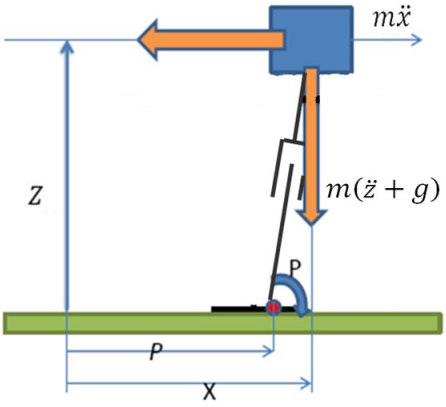

Figure 3.7: A schematic view of the inverted pendulum in XZ plane

In the sagittal plane, the horizontal and vertical positions of CoM are denoted by x and z, respectively. Gravity g, horizontal CoM acceleration $\ddot{x}$, and vertical CoM acceleration $\ddot{z}$, create a moment T_p around the center of pressure (CoP) point P_x. Equation (3.4) provides the moment around P.

$$T_p = M(g+\ddot{z})(x-P_x) - M\ddot{x}z \tag{3.4}$$

We know from [11] that when the robot is dynamically balanced, ZMP and CoP are identical, therefore, the amount of moment in the CoP point must be zero, $T_p = 0$. By assuming the left hand side of equation (3.4) to be zero, equation (3.5) provides the position of the ZMP based on the position and acceleration of CoM. In order to generate a 3D walking, the CoM must also move in the frontal plane; hence, another inverted pendulum must be used in y direction. Using the same assumptions, equation (3.6) is given for movements in the frontal plane.

$$P_x = x - \frac{z}{g+\ddot{z}}\ddot{x} \tag{3.5}$$

$$P_y = y - \frac{z}{g+\ddot{z}}\ddot{y} \tag{3.6}$$

It is interesting that by considering the inverted pendulum model assumptions on ZMP equations (2.5) and (2.6), equations (3.5) and (3.6) can be derived. By considering all masses to be concentrated on trunk of the robot, equation (3.7) is obtained from the equations (2.5) and (2.6).

$$p_x = \frac{m(\ddot{z}+g) - m\ddot{x}z - I_y\ddot{\Omega}_y}{(\ddot{z}_i+g)m} \tag{3.7}$$

Since the effect of the moment of inertia I_y and angular accelerations Ω_y are negligible [62], we can drop the last term in the equation (3.7), thus the same equation presented in (3.5) and (3.6) will be resulted.

In order to apply the inverted pendulum model in a biped walking approach, first the positions of the support foot during a walk must be determined. Then, the ZMP trajectory is designed based on support foot positions. The vertical CoM position and acceleration trajectory must also be determined as the input to the inverted pendulum model. In the final step, the horizontal CoM position is calculated by solving the inverted pendulums' differential equations that were given in (3.5) and (3.6). Finally, an inverse kinematics method is used to generate the angular trajectories of each joint based on the planned position of the feet and generated CoM position. The servomotors are used to control the joints position to follow the generated angular trajectories.

The main issue of using the inverted pendulum is on how to solve inverted pendulum's differential equations. Kagami et al. presented a numerical approach to solve the differential equations by discretizing them [65]. This approach is more suitable for generating more static walking than dynamic walking; we present an approach in which the CoM is generated in a more dynamic manner [25]. We explain this approach in Chapter 7. Another issue of using inverted pendulum model is how to generate the CoM vertical trajectory, usually researchers use heuristic approaches in the modelling of CoM height trajectory. We present optimization scenarios to obtain optimized CoM height trajectories in different walks' characteristics, such as energy consumption and walk speed. In Chapter 7, we explain these optimization scenarios along with their results.

3.2.3 Summary

There are a couple of drawbacks to ZMP based approaches. First, the ZMP methods do not account for non-level surfaces or unexpected impact forces. Second, walk generated by ZMP based approaches are not energy efficient, since these approaches control actuators actively to maintain a continuous ZMP trajectory inside the support polygon. This leads to not make use of the natural dynamics of a humanoid, and generate a non-human looking gait.

In order to apply ZMP dynamics on designing a feed-forward control policy for controlling the walk's stability, the ZMP dynamics should be simplified. Respecting this, concentrated-mass models are used to simplify and approximate ZMP dynamics. The advantage of using simplified models is that they enable us to use the ZMP concept in a walk generation task. However, models' simplification has also an inherent disadvantage that the actual ZMP is not used, and sometimes the different between the approximated and actual ZMP is big that makes the robot unstable. Consequently, approaches used these models often try to create a walk in a

conservative way. As an example, they try to maintain the ZMP position in the center of the support polygon. Both Cart-Table model and inverted pendulum model have this issue.

Cart-Table model the ZMP dynamics with big simplifications, this results to have motion problems [81], such as bent knee and constant CoM height. From the viewpoint of energy efficiency, the walk generated using Cart-Table model are energy inefficient since the high knee torque must be applied to maintain a bent-knee posture continuously. The constant CoM height does not allow the robot to have big steps. In the Figure 3.8, leg in bent knee position is shown.

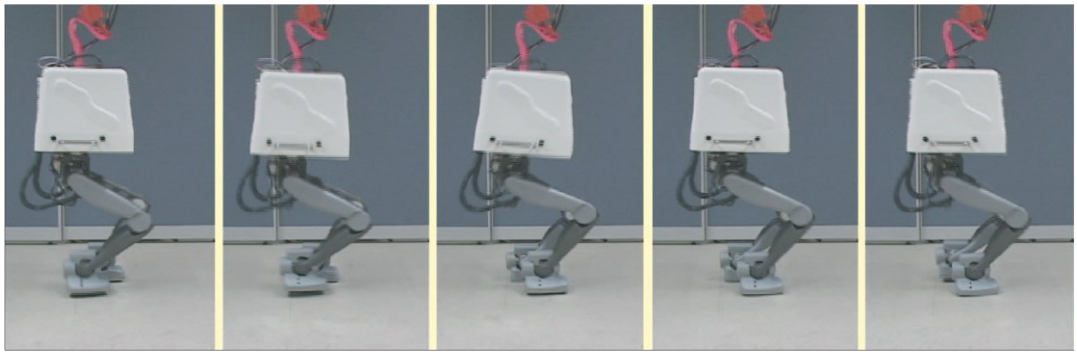

Figure 3.8: Bent knee position

Inverted pendulum model remedy the bent-knee and constant height issues, but many parts of its development approach is still under discussion, including, which CoM vertical trajectories can be chosen as the input to the inverted pendulum model, and how to solve its differential equations. In Chapter 7, we will answer to these questions.

As a conclusion, despite the drawbacks, ZMP based approaches still are the dominant approach to generate stable walking, since they are developed based on well-established walk stability analysis and controller. The most popular humanoid robots also utilize some variation of ZMP based approaches to perform their locomotion, including Honda ASIMO [45], Aldebaran's NAO [46], HPR-3 [3] and Hubo [47].

3.3 Bio-inspired Central Pattern Generator

In this section, biologically inspired robot locomotion approaches based on locomotors Central Pattern Generators (CPGs) [82] will be explained. From neurobiological studies viewpoint, central pattern generators are neural networks capable of producing coordinated patterns of rhythmic activity without any rhythmic inputs from sensory feedback or from higher control centers, which underlie many rhythmic behaviors both in invertebrate and vertebrate animals [83]. There is clear evidence showing that rhythmic movements are generated centrally without requiring sensory information in biological systems. Researchers could extract and cut

off the spinal cord of a primitive fish form the body, and it could still produce locomotion rhythms, they called it fictive locomotion [83]. Similar fictive locomotion has been reported in salamander or in many other animals [84]. Although sensory feedback is not needed for generating the rhythms, it plays an important role in coordinating CPGs and body movements with each other and shaping produced rhythmic patterns [85]. Several experiments demonstrate this important fact, i.e. some studies show that the tail moving of the lamprey will be influenced by the other CPGs activities [86]. Duysens predicted the possibility of the CPG's existence in humankind and probably CPGs has some role in human walking [85]. According to this biological background, it is interesting to study mathematical models of CPGs and their possible applications in humanoid walking approaches.

Depending on types and goals of the study and the level of abstraction that needs to be investigated, CPGs are designed and modeled by many different approaches. Main approaches can be categorized in two big subgroups; connectionist models, and nonlinear oscillator models.

Connectionist models use simplified neuron models that are interconnected with each other as a neural network such as leaky-integrator neurons or integrate-and-fire neurons [87]. These approaches study how rhythmic activities are generated using network properties. A drawback to these approaches is that it is hard to discover the relation between a neuron's parameters and produced rhythmic patterns; consequently, it is difficult to design the interconnection and feedback pathways of the neural oscillators.

Nonlinear oscillator models use sets of differential equations as a mathematical model of an oscillator and attempt to model inter oscillator coupling [82]. In this case, an oscillator represents the activity of a complete oscillatory center instead of a single neuron or a small circuit. In legged robotics, researchers try to use a population of oscillatory centers and model inter-oscillator couplings to study how it affects the synchronization and the phase of legs. Gait patterns are generated through the phase coupling of oscillators, and by changing the coupling methods among oscillators, different gait patterns can be produced.

Researchers often use oscillatory centers that are well established in dynamical system theory. Zielinska et al. used coupled Van der Pol oscillators to generate rhythm locomotion control signals for a two-legged walking machine [88]. Filho et al. used mutually coupled Rayleigh oscillators and Van der Pol's oscillators to generate control signals that were similar to human locomotion [89]. A Phase oscillator based on Kuramoto's model is also used to construct the CPG model [90]. Matsuoka proposed a network of oscillators that used a set of inhibitory connected neuron oscillators to build a CPG model [49]. All of these oscillators have a fixed waveform for a given frequency and do not have the capacity for learning and adaptation according to the input signals. Righetti et al. presented a programmable CPGs model based on

Hopf oscillator and applied it to the locomotion control of humanoid and quadruped robots [91]. This CPGs network could learn rhythmic input signals and learning process was totally embedded in the dynamics of the system. Several interesting properties make CPG models, especially programmable CPGs, useful for generating walking motions. We will fully explain programmable CPGs, and try to address the possible advantages of using them.

In section 3.3.1, we will introduce the dynamics of Hopf oscillator, and then the dynamics of other interested oscillators will be discussed. In section 3.2.2, we will explain learning and adaptation methods based on the Hopf oscillator to construct programmable CPGs. In section 3.3.3, implementation of CPGs on the humanoid robot will be introduced. Finally, we will talk about pros and cons of using programmable CPGs and possible application in biped locomotion approaches that can be done in this field.

3.3.1 Hopf Oscillator

Hopf oscillator is one of the dynamical systems that is used in CPGs to produce the walking trajectories [92]. Hopf concept is studied in the field of nonlinear dynamics and Bifurcations. To explain the concept of Hopf oscillator first we address the concept of supercritical Hopf bifurcation.

Supercritical Hopf bifurcations can occur in two dimensional phase space or any phase spaces bigger than two dimensions. In equations (3.8) and (3.9), a general form of a supercritical Hopf bifurcation is given, in which a two dimensional nonlinear system is used. They are presented in polar coordinates (r, θ).

$$\dot{r} = \mu r - r^3 \tag{3.8}$$

$$\dot{\theta} = \omega - br^2 \tag{3.9}$$

where μ controls the stability of the fixed point at the origin, ω gives the frequency of infinitesimal oscillations, and *b* determines the dependence of frequency to the amplitude.

In the Hopf oscillator, parameter *b* is assumed to be zero and the radial motion (presented in equation (3.8)) defines the amplitude of the produced trajectory. Analyzing the system in this polar form is easier since the radial and angular motions are independent.

For this two-dimensional nonlinear system, finding the trajectories by solving the differential equations analytically is typically not possible. Even when explicit formulas are available, it is too complicated to provide much insight. Indeed, it is better to try to determine the qualitative behavior of the solutions by drawing the phase portrait. Phase portrait of a system shows the overall picture of trajectories in phase space.

In terms of the flow in phase space, a supercritical Hopf bifurcation happens when a stable spiral changes to an unstable spiral surrounded by a small, nearly elliptical limit cycle [93]. These qualitative changing portraits are called bifurcations. An isolated closed trajectory in phase portrait is also called a limit cycle. Isolated means that neighboring trajectories are not closed; they spiral either toward or away from the limit cycle [93]. In the case of spiral toward the limit cycle, all neighboring trajectories approach the limit cycle and it is called stable limit cycle or attractors. Stable limit cycles are very important scientifically since they model systems that exhibit self-sustained oscillations. Hopf oscillator has a stable limit cycle and is a self-sustained oscillator.

Major benefits of the self-sustained oscillators are summarized in its two characteristics. First, these systems oscillate even in the absence of external periodic forcing. Second, the oscillator is robust in the face of the external input forces which perturb the system slightly, since the generated trajectory spirals toward the limit cycle and the system returns to the standard cycle.

In the Hopf oscillator given in (3.8) and (3.9), the value of the parameter μ controls the bifurcation and the value at which the change occurs are called bifurcation points. The key point of the dynamics of this system is defined by the radial motion (3.8). Since parameter b is assumed to be zero, the angular motion (3.9) explains that all trajectories rotate around the origin with constant angular velocity $\dot{\theta} = \omega$. To analyze the dynamics of the system, calculating the root of the radial motion has an important rule; it shows where fixed points in the flow of the amplitude motion equation will be.

In equation (3.8), roots of the radial equation or positions of its fixed points can be controlled by the value of the parameter of μ, where bifurcation point is $\mu = 0$. For $\mu < 0$, the root of the radial motion will be zero, and phase portrait is the stable spiral with the origin of $r = 0$ whose sense of rotation depends on the sign of ω. For $\mu > 0$, root of the dynamics of the radial equation will be $\sqrt{\mu}$ and 0, therefore its phase portrait shows an unstable spiral at the origin and the stable circular limit cycle at $r = \sqrt{\mu}$. Figure 3.9 shows the different phase portraits and generated trajectories for different values of μ around the bifurcation point.

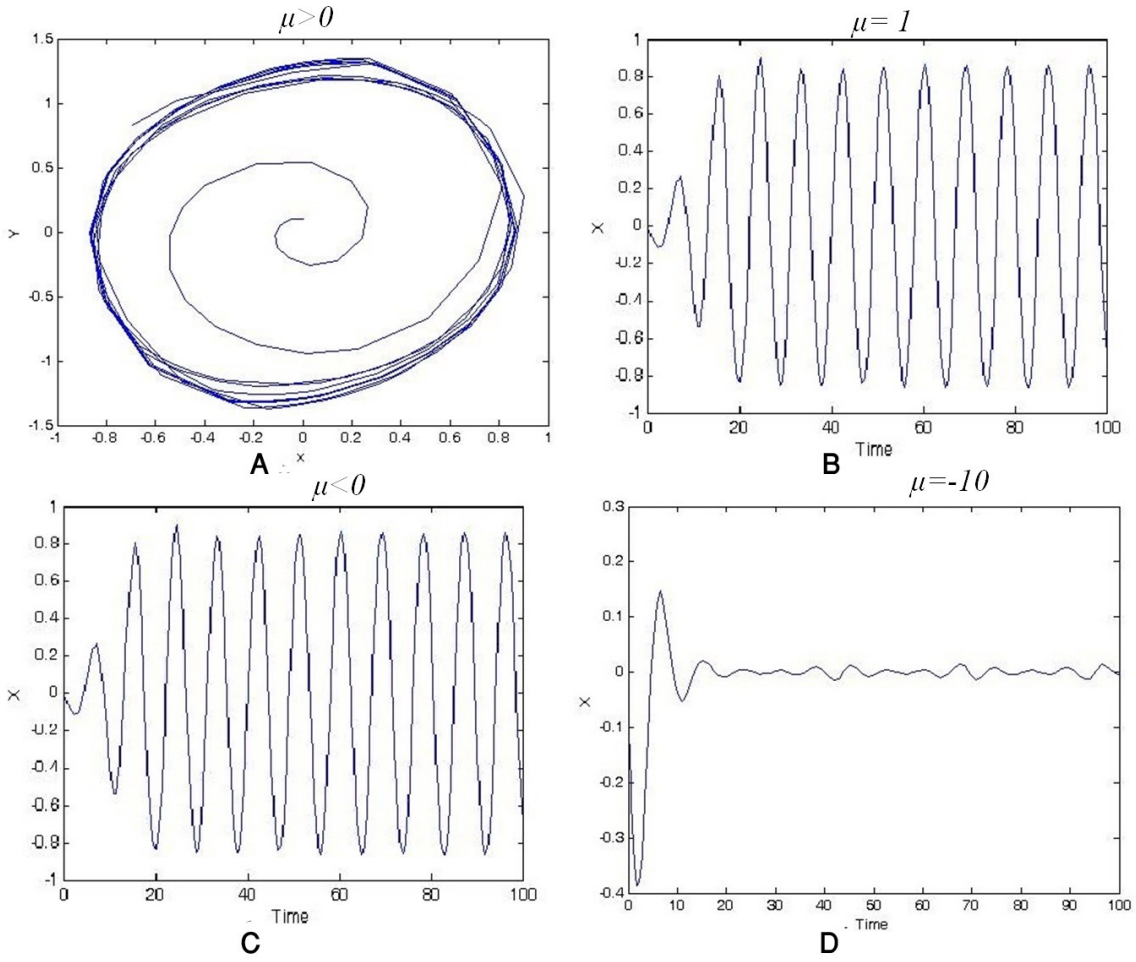

Figure 3.9: A) Phase portraits around the bifurcation point while $\mu>0$ B) Trajectory generated by the oscillator while $\mu= 1$ C) Phase portraits around the bifurcation point while $\mu<0$ D) Trajectory generated by the oscillator while $\mu=-10$

The Hopf oscillator, given in (3.8) and (3.9), can be rewritten in Cartesian coordinates by considering $x = r\,cos\,\theta,\ y = rsin\,\theta$, equations (3.10) and (3.11) describe this oscillator.

$$\begin{aligned}\dot{x} &= \dot{r}\cos\theta - r\dot{\theta}\sin\theta \\ &= (\mu r - r^3)\cos\theta - r\omega\sin\theta \\ &= (\mu - r^2)\,x - \omega y\end{aligned} \tag{3.10}$$

$$\begin{aligned}\dot{y} &= \dot{r}\sin\theta + r\dot{\theta}\cos\theta \\ &= (\mu r - r^3)\sin\theta + r\omega\cos\theta \\ &= (\mu - r^2)\,y + \omega x,\end{aligned} \tag{3.11}$$

where $r = \sqrt{x^2 + y^2}$, $\mu > 0$ determines amplitude of the produced trajectory and ω controls the frequency of the oscillator. As it was mentioned, in phase space, this oscillator has a stable limit cycle with radius $\sqrt{\mu}$ and angular velocity ω rad/sec.

Hopf oscillator can be coupled with a periodic signal *F* and perturbed by this signal. In equation (3.12), A coupled Hopf oscillator is given. Since Hopf oscillator has stable limit cycle, small perturbations around its limit cycle do not change the general behavior of the system. $\varepsilon > 0$ controls the preturburation size.

$$\begin{aligned} \dot{x} &= (\mu - r^2)\, x - \omega y + \epsilon F(t) \\ \dot{y} &= (\mu - r^2)\, y + \omega x \end{aligned} \tag{3.12}$$

In the next section, we will describe an existing method that enables the model to adapt itself, in order to synchronize with big perturbation caused by periodic input signal *F*.

3.3.2 Synchronization and Learning Rule

Nonlinear oscillators are interesting because of their synchronization properties when they are coupled with other oscillators or with an external input signal. Most studies use phase-locking behavior for their coupling method [94]. According to equation (3.12), relying on the external perturbation and the state of the oscillator, i.e. the position of the point on the limit cycle, the perturbation accelerates the phase point or slows it down. Since the perturbation is a periodic signal, the average of this acceleration or deceleration depends on the frequency difference [91]. If intrinsic frequency of the oscillator is close to the frequency component of the external perturbation, phase-lock behavior will appear, and synchronization will be done perfectly.

In 2006, Righeti et al. designed an adaptive oscillator based on Hopf oscillator which was able to learn CPGs frequency from the frequency of periodic input signals [91]. They called their adaptive mechanism dynamic Hebbian learning because it shared similarities with correlation-based learning found in neural networks [95].

Considering equation (3.12) that represents Hopf model coupled with periodic input signal or *F*, the system can be rewritten in polar coordinates by equation (3.13). In order to tune the frequency of the oscillator, the frequency should evolve toward the frequency of the input signal. It was shown that if the effect of perturbation on the dynamics of the frequency is the same as the effect of the perturbation on the angular motion, it causes (on average) driving ω toward the frequency of the perturbation [91]. Therefore, dynamics of the system on frequency is presented by the equation (3.14).

$$\begin{aligned} \dot{r} &= \mu r - r^3 + \varepsilon F \cos\theta \\ \dot{\theta} &= \omega - \frac{\epsilon}{r} F \sin\theta \end{aligned} \tag{3.13}$$

$$\dot{\omega} = -\epsilon F \sin\theta \tag{3.14}$$

Equation (3.14) leads the system to learn the frequency of the periodic input signal and the oscillator can be synchronized to it by the evolution of time, where ϵ and ε are coupling constants. Equation (3.15) represents the frequency dynamics in Cartesian space.

$$\dot{\omega} = -\epsilon F \frac{y}{\sqrt{x^2 + y^2}} \tag{3.15}$$

The presented equations (3.13) to (3.15) are mostly adequate to be utilized for generating an input signal that consists of one frequency compound. Reghetti et al. extended this approach in order to use the network of the coupled oscillators for learning the periodic teaching signal that has the different frequency components [96]. Since the oscillation of the Hopf oscillator is harmonic, like Fourier series representation, proper linear combination of several Hopf oscillators can produce any periodic input signals. The structure of the network of adaptive Hopf oscillators is shown in the Figure 3.10.

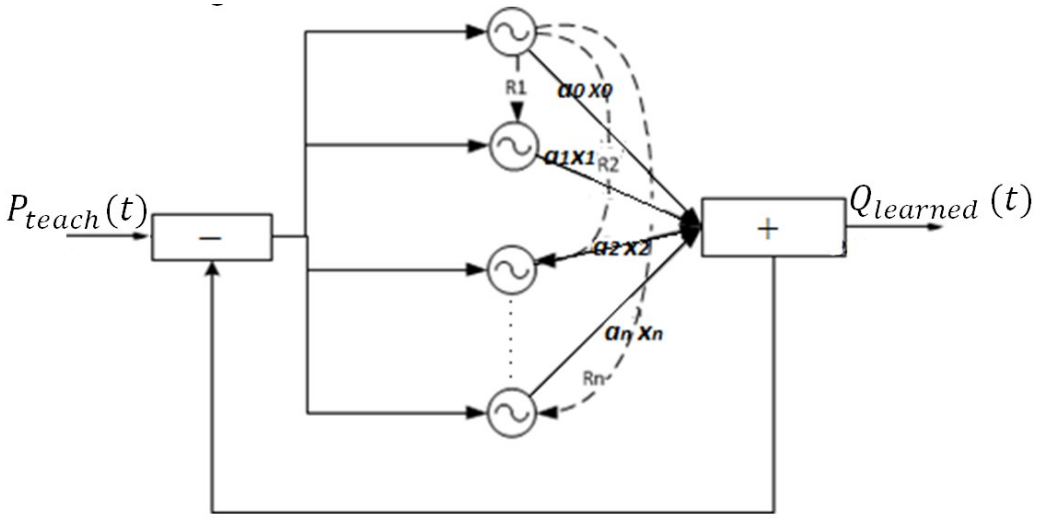

Figure 3.10: A Schematic view of network of adaptive Hopf oscillators as a CPG block

Each adaptive oscillator is responsible for learning one frequency component of the signal. The network can be designed by *N* oscillators and each oscillator is denoted by *i*. The learning of the frequencies is done by the dynamic Hebbian learning which was presented in (3.15). The output of the system is the weighed sum of the output of the oscillators, $Q_{learned}(t) = \sum_i a_i x_i$, here a_i is assumed the amplitude of each learned frequency.

$F(t) = P_{teach}(t) - Q_{learned}(t)$ is a negative feedback loop, which the already learned frequencies is subtracted from the teaching signal. This leads the system to adapt to the remaining frequencies component that have not yet converged. To enable each oscillator to have its own phase shift, a variable encoding phase difference between the oscillator and the first oscillator of the network is associated with each of them. In order to reproduce any phase

relationship between the oscillators, the Kuramoto coupling scheme [90] is used. The equations describing the total dynamics and learning of CPGs are given as follows:

$$\dot{x_i} = \gamma(\mu - {r_i}^2)x_i - \omega_i y_i + \epsilon F(t) + \tau \sin(\theta_i - \emptyset_i) \tag{3.16}$$

$$\dot{y_i} = \gamma(\mu - {r_i}^2)y_i - \omega_i x_i \tag{3.17}$$

$$\dot{\omega_i} = \epsilon F(t)\frac{y_i}{r_i} \tag{3.18}$$

$$\dot{a_i} = \beta x_i F(t) \tag{3.19}$$

$$\dot{\emptyset_i} = sin(\frac{\omega_i}{\omega_0}\theta_0 - \theta_i - \emptyset_i) \tag{3.20}$$

$$\theta_i = sgn(x_i)cos^{-1}(-\frac{y_i}{r_i}) \tag{3.21}$$

Equations (3.16), (3.17) and (3.18) are representing Hopf oscillator and its frequency learning, where γ controls the speed of recovery after perturbation, and the other parameters have the same description that were given in equations (3.12) and (3.15). In equation (3.16) Kuramoto coupling method is represented by $\tau\, sin(\theta_i - \emptyset_i)$ in order to achieve phase synchronization between oscillators, in which each adaptive oscillator is coupled with the first oscillator *i=0*, with strength τ to keep correct phase relationships between oscillators. $\emptyset_i$ is the phase difference between oscillator *i* and 0.

Equations (3.20), (3.21) show how $\emptyset_i$ converge to the phase difference between the phase of the first oscillator θ_0 scaled at frequency ω_i ,and the phase of i^{th} oscillator θ_i. The learning rule for updating a_i is presented by equation (3.19), where β is learning rate. These learning rules show how correlation between x_i and *F(t)* will be maximized. Correlation will be positive in average and will stop increasing when frequency component ω_i disappears from *F(t)* because of the negative feedback loop. The negative feedback is working as a measure of the amount of the error, and learning rule is working like the perceptron rule and since the input signal is linearly separable, the above online algorithm will converge.

As conclusion, applying learning rules given as differential equations, parameters such as intrinsic frequencies, amplitudes, and weights of phase coupling can be automatically adapted to a teaching signal. This approach is more applicable compared to a classical Fourier transformation in representing a periodic signals because the oscillators can have any phase relationship and not only 0, π differences. Moreover, one of its interesting aspects is that the learning is completely embedded in to the dynamical system, and does not require external optimization algorithms. In the next section, we will explain the possible applications of CPGs in humanoid walking approaches.

3.3.3 CPGs Applications on Biped Walking

Central pattern generator based on Hopf is well-designed oscillator and it has been applied on humanoid robots for the task of controlling gaits [96] [97]. As it was mentioned in previous section, CPGs offer multiple interesting features mainly caused by the stability properties of the limit cycle behavior.

In biped locomotion approaches using programmable CPGs, gait generators produce angular trajectories of each joint as input trajectories for CPGs, and output learned trajectories are controlled by the PID of each joint. The block of the gait generator is not studied in this section. Gaits can be generated by sine based approaches which will be explained in the section 3.4 [98].

Although each joint has its own CPG, as it will be presented in the section 3.4, the right leg and the left leg have the π phase shift difference and all of the angular trajectories of the joints have equal frequency. Therefore, each block of the CPGs for each joint also can be coupled. Structure of CPGs which was designed for a humanoid is shown in Figure 3.11. This system is able to generate learned trajectories for all of joints, and it can use the benefit of properties of the CPGs, such as limit cycle behavior and adaption capabilities.

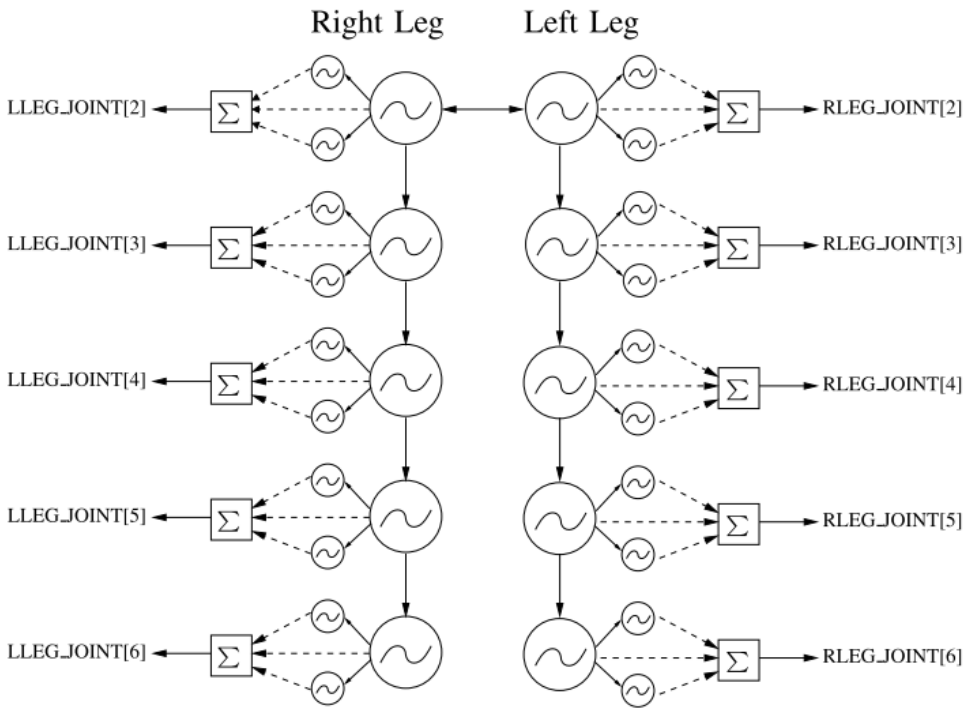

Figure 3.11: Network of CPGs applied on a humanoid [96]

Modifying amplitude and frequency of the input trajectories or even any change in the input trajectories can cause a smooth change in the learned trajectory. The smooth modulation and change of the gait of the humanoid can improve the basin of the stability of humanoid walking. This intrinsic property of the CPGs, can improve the stability of start of the walk [99]. Another

advantage of using CPGs is that the robot can have a modulation like changing its own speed. Therefore, it allows the implantation of a higher level control easier, for example designing path planning procedure will be easier.

Another benefit of using the CPGs is that sensory feedback can be added to the oscillator mechanism. If additional components like feedback information are introduced in the equation (3.15), the oscillator will smoothly incorporate this "perturbation" and once it is gone, converge back to its original configuration. However, it is hard to organize feedbacks to represent the dynamics of body and design feedback loops.

One of the usual feedback pathways introduced in the literature, is for maintaining lateral stability during locomotion [99]. This feedback pathway is inspired by vestibular system in humans that measures the tilt of the body and activates contralateral muscle to keep balance [96]. In this sense, Gyroscopes located in the chest of the robot are used to calculate the lateral tilt of the body. When this tilt is increasing, the robot should tilt in the opposite lateral direction.

There are 2 DoFs controlling the lateral direction in the robot, the hip and the ankle joint that move in sagittal plane. The feedback pathways built inside the CPGs for controlling these joints and maintaining lateral stability during locomotion was introduced in [96]. The equation (3.22) presents this feedback pathway based on the equations (3.16) and (3.17).

$$\begin{aligned} \dot{x}_i &= \gamma(\mu - r^2)x_i - \omega_i y_i + \tau\, sin(\theta_i - \emptyset_i) + g_k \frac{x_i}{r_i} \\ \dot{y}_i &= \gamma(\mu - r_i^2)y_i - \omega_i x_i + g_k \frac{y_i}{r_i} \end{aligned} \tag{3.22}$$

g_k is the gain that is calculated by $g_k = K_{lateral} \cdot \varphi_{lateral}$, where $\varphi_{lateral}$ is the lateral tilt of the body, and $K_{lateral}$ is the same for both feedback pathways of the two joint and it is the same for both legs, resulting in a symmetric change of trajectory for both feet. These feedbacks on the radius of the limit cycle of all oscillators associated to the hip and ankle joints will be projected, since only the amplitude of the trajectories interested. This projection is presented to be sure that the phase is preserved. As conclusion, using CPGs can prepare online modulation of trajectories on humanoid robots and by using sensory feedback, which can increase basin of stability of the gait.

3.3.4 Summary

In conclusion, CPG approaches are categorized as model-free approaches, therefore they do not need to consider the physical model of the robot. There are several interesting properties that make CPG models useful for the control of locomotion in robots compared to alternative model

free approaches such as sine-generators that will be discussed in section 3.4, some advantages of using CPG approaches are the following:

- They are robust against the perturbation because of having stable limit cycle behavior
- They have easy modulation (changing the speed of walking)
- Sensory feedback can be added to oscillators

Despite of the fact that CPG is a well-defined approach, it has still not been implemented on a real humanoid robot to generate an omnidirectional walk. Since an omnidirectional walk contains different walk directions, the robot should have the ability to change them in real-time. This needs a proper feedback from the dynamics of the walk, which is very hard to organize sensory feedback to represent the dynamics of the body. CPGs still do not have a defined methodology to design the feedback pathway from the stability indicators such as ZMP, CoP.

ZMP based approaches presented in section 3.2 can generate omnidirectional walking, but they usually generate their joints angular trajectories by using inverse kinematics, in this case CPG cannot be employed to generate joints' angular trajectories. CPGs can be used to generate other walking trajectories than joints' angular trajectories. We used the programmable CPGs to model CoM vertical trajectories in a ZMP based walking approach [27] [25]. This approach will be fully explained in Chapter 7.

3.4 Truncated Fourier Series

Bipedal walking as a complex motion, involves most of humanoid robot's joints to produce gaits. A gait can be assumed as a cyclic, periodic motion of the joints of a legged robot, which requires the sequencing or coordination of the legs to obtain reliable locomotion. There is the temporal and spatial relationship between all the moving parts of a legged robot [48].

It was shown that gaits can be modeled by the sine based formulation like Fourier series. In 2008, the Partial Fourier Series (PFS) method [100] was introduced for generating a forward walk motion in the sagittal plane. The PFS method is easy to implement on humanoid robots, and interesting results have been observed for a forward walk motion in [100].

Researchers attempt to imitate the human walking style [101][102]. Therefore analyzing human walk pattern has been used for acquiring beneficial information about this motion. Human walk has been investigated from many angles; walking trajectory is one of them. The walking trajectory is divided into several types. Positional trajectory and angular trajectory are two of them. In angular trajectory, the angle of each joint is plotted at a certain time slice.

In 2007, Yang et al. presented a model-free approach based on Truncated Fourier Series (TFS) for generating angular trajectories of an stable bipedal locomotion [48]. They called it the Genetic Algorithm Optimized Fourier Series Formulation (GAOFSF) and it was inspired by the shape of the angular trajectories of a human walking. In this approach, a Truncated Fourier Series (TFS) formulation were used, in which its coefficients were determined and optimized by a Genetic Algorithm (GA) to generate human-like stable forward walk on flat terrains as well as on slopes. The ZMP stability indicator was used to prove that TFS can generate suitable angular trajectories for stable biped walking. It was interesting that the walk was generated only by joints' movement in sagittal plane. In 2009, Shafii et al. presented an optimized gait generator based on TFS approach, which was implemented in a simulated NAO humanoid robot [103]. In another study, TFS parameters are reduced by 2 dimensions [104].

TFS based approach does not require inverse kinematics and stable gaits with different step lengths and stride frequencies can be readily generated by changing the value of only one parameter [48]. We will use this approach in our proposed model-free gait generator. To explain TFS approaches deeply, basic TFS concept is introduced in the section 3.4.1. TFS application on generating walking movement will be explained fully in the section 3.4.2. Shafii et al. also extended the basic TFS enabling the generation of arm angular trajectories that provide smooth and robust walking [105]. The arm angular trajectory generation will be presented in section 3.4.3.

3.4.1 Basic Concept

Biped angular trajectory of two joints; hip and knee captured from human walking are shown in Figure 3.13[48]. By capturing the main features of Figure 3.13 and giving a general form to make it applicable to robots Figure 3.12 is drawn.

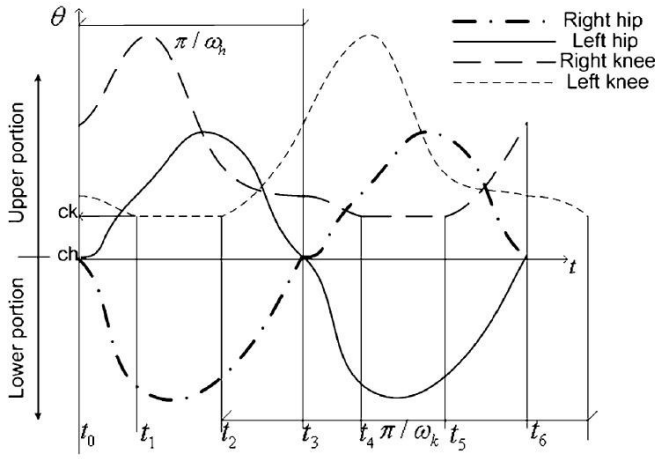

Figure 3.12: Gaits elaborated from human gaits features [48]

In Figure 3.12, the angle of hip and knee joints in one period of walking signal from t_0 to t_6 is represented. In time $[t_0, t_2]$ and $[t_5, t_6]$ the left leg is support leg and the right leg is swing leg, but in time range $[t_2, t_5]$ the right and left legs play the role of support and swing legs, respectively. In another word, in two times of t_2 and t_5 the roles of the two legs are switched with each other. At time t_3 where two hip trajectories intersect, two thighs cross each other.

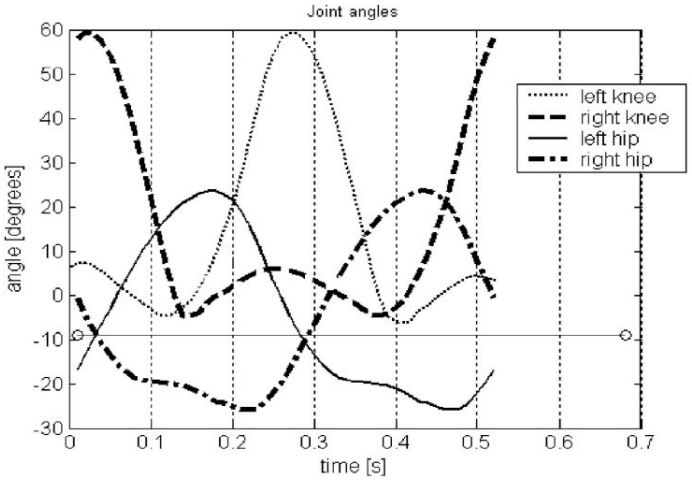

Figure 3.13: Human walking angular trajectory[48]

Considering the fact that all joint trajectories of human walking are periodic and similar to sine or cosine signals [106], the generation of these angular signals can be done by Fourier series.

The original definition of Fourier series is described by the following formula:

$$F(t) = \frac{a_0}{2} + \sum_{i=1} (a_i \, cos \frac{i\pi t}{L} + b_i \, sin \frac{i\pi t}{L}) \tag{3.23}$$

The first term $(a_0/2)$ of equation (3.24) represents the DC bias of the signal, and *L* represents half of the largest period that exists in the signal. By $\omega = \pi/L$ then the frequency form of Fourier series is achieved as follows:

$$F(t) = \frac{a_0}{2} + \sum_{i=1} (a_i \, cos(i\omega t) + b_i \, sin(i\omega t)) \tag{3.24}$$

Here, ω is the frequency of the periodic signal. Any complicated signal can be produced by this formula when i is considered infinite. However, when the value of i is limited to a finite number, precision of generated signal is reduced and this type of Fourier series is called partial sum of the Fourier series. According to Figure 3.13, Human walking angular trajectories are too complicated to be produced by a finite Fourier series band limited to the second harmonic. Therefore a modified finite Fourier series as a Truncated Fourier series (TFS) is used in the modeling of the joints' angular trajectories.

3.4.2 Modeling of Legs Movement

According to Figure 3.12, the legs angular trajectories are divided to two parts; upper portion and lower portion. Whereby each portion can be assumed as an odd function, the cosine part of TFS is eliminated. Therefore, the TFS is reduced to equation (3.25) to generate each portion of trajectory.

$$F(t) = a + \sum_{i=1}^{n} b_i \, sin(i\omega t) \tag{3.25}$$

Where ω is the fundamental frequency of signal and *a* is signal offset. Separate production of each portion, generates signals with different upper and lower portions. The number of parameters for generating these signals is also less than the parameters used in Fourier series. As shown in Figure 3.13, each signal has an offset, C_h and C_k are hip trajectory and knee trajectory offsets respectively.

From t_0 to t_2 , the left leg is considered as supporting leg and the variation of its knee angle is so small that it can be assumed fixed. This duration of walking is named lock phase. In addition, the amount of shift phase of the two leg trajectories signal is half of the period of each signal. The trajectories for both legs are identical in shape but shift half of the walking period in time. Therefore by figuring out walking angular trajectory of one leg the other leg trajectory is obtained. Using equation (3.25) and considering curves of Figure 3.13, the TFS for generating each portion of hip and knee trajectories are formulated as follows:

$$\theta_k^+ = \sum_{i=1}^{n} C_i \sin(i\omega_k t_2) + C_k \, , \omega_k = \frac{2\pi}{T_K} \qquad \theta_k^- = C_k \geq 0 \tag{3.26}$$

$$\theta_h^+ = \sum_{i=1}^{n} A_i \sin(i\omega_h t_2) + C_h \ , \omega_h = \omega_k \tag{3.27}$$

$$\theta_h^- = \sum_{i=1}^{n} B_i \, sin(i\omega_h t_6) + C_h \ , \omega_h = \omega_k \tag{3.28}$$

In these equations, the plus (+) sign represents the upper portion of walking trajectory and the minus (-) shows the lower portion. A_i, B_i and C_i are constant coefficients for generating signals. The h and k index stands for hip and knee respectively. C_h and C_k are signal offsets and T_k is assumed as period of knee trajectory. Considering the fact that all joints in walking motion have equal movement frequency [38], the equation $\omega_k = \omega_h = 2\pi/T_k$ can be derived. Parameter t_3 shows the end time of hip trajectory in upper portion and starts its down portion, t_6 shows the end time in down portion. These parameters are not significant since they can be obtained when the hip trajectory intersects the C_h line. However, parameter t_2 represents the end time of knee

lock phase and must be considered to produce knee trajectory. Therefore Truncated Fourier series parameters to produce trajectories are; C_h, C_k, A_i, B_i, C_i, t_2, t_6 and ω_k. In [48], it was shown that by finding the proper values of these 8 parameters, the proper legs movements in sagittal plane can be found, and by using only these movements in sagittal plane an stable forward walk can be generated.

In 2009, Shafii et al. presented an approach to reduce the parameters of this TFS formulation to 6 parameters [104]. This reduction was achieved by analysing the relationship between human joints and its angular trajectories during a walking cycle. Figure 3.14 shows the angular relationship between knee and hip through human walk [107].

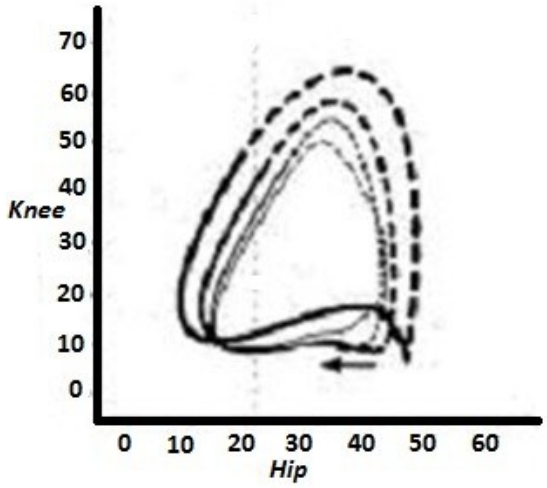

Figure 3.14: Hip knee diagram in human walk [107]

We can get some useful information about knee lock phase by looking at Figure 3.14. According to this figure, the straight line represents the knee lock phase and this phase starts when the hip angle reaches its maximum value. With the same logic, the lock phase is ended with minimum value of hip angle.

So by specifying the start and end time of lock phase, two parameters of (t_1, t_2) could be eliminated. Thus, the number of variable for optimization decreased to 6. In this case, an optimization algorithm is needed to optimize a 6 dimension problem for finding the best gait generator. This omission has many advantages such as: reducing the search space of optimization problem and increasing the convergence speed of optimization approach.

3.4.3 Modeling of Arm Movement

Humans swing their arms naturally when they run or walk. Even though arm swing has often been compared with pendulum motion, arm motion is not a completely passive phenomenon.

Muscle activity controls arm swing magnitude and timing while human walk [108]. As humans change walking speed, their nervous systems adapt muscle activation patterns to modify arm swinging for the appropriate frequency. Humans have neural connections between their upper limbs and lower limbs, which coordinate muscle activation patterns during locomotors tasks. Mechanical analysis indicates that arm swing during human locomotion helps to stabilize rotational body motion [109].

Elftman first proposed that arm swing during walking balances torso torques caused by swinging of the lower limbs [106]. This idea has been studied further by others with the same general conclusions [108], [110]. The primary mechanical effect of arm swing during the walking reduces body-twisting torque along the vertical axis. The upper limb moves forward when the contra lateral lower limb moves forward; therefore, the angular momentum of the contra lateral upper and lower limbs partially balance each other, reducing the rotational moment between the foot and the ground [110].

In case of straight forward walking, especially on high speed walking, human beings must swing arms to absorb yaw moment caused by legs swing. The role of arm swing is vital on this problem [111]. Figure 3.15 shows yaw moments caused by legs swing and how hand will absorb the yaw moment. Finally, we can have a robot with faster walking in a stable manner by swing arms.

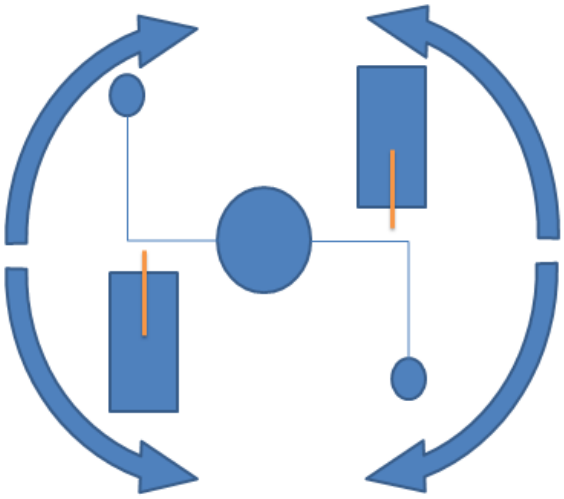

Figure 3.15: The role of hand in absorbing yaw moment

During human walking, the arms normally swing in an opposite manner to legs, which helps to balance the angular momentum generated in the lower body [106], [108]. Humans swing their arms close to 180º out of phase with their respective legs during walking [13], Figure 3.16 shows the trajectory of legs and arm swings and the relation between them in a straight human walking [106]. It is shown that trajectory of arms is similar to sinusoidal signal with same

frequency of legs.It is found that the speed of walking has a strong effect on arm swing during gait. By increasing gait's speed, the arms may swing higher and faster to reduce the effects of longer, quicker steps by the legs [112].

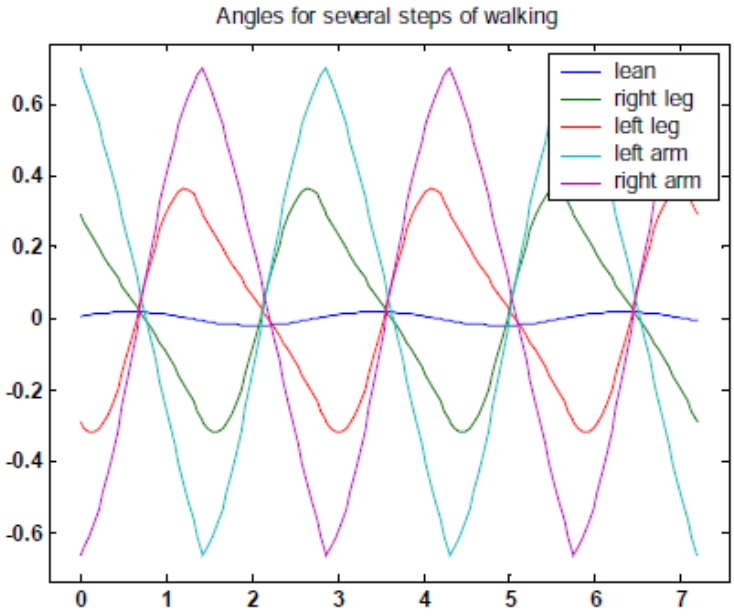

Figure 3.16: Trajectories of legs and arms [106]

It can be expected that the utilization of arm swing provides good performance to yaw moment stability, and recovery from stumbling. The effectiveness of this method was confirmed as an improvement of accuracy of straight walking in different speeds [105]. As has been shown, the trajectory of arms is a sinusoidal signal; therefore, to produce the proper signal to arms swing, it is enough to obtain adequate parameters for equation (3.29).

$$F(t) = A\,sin(\omega_{arm} t) + b \tag{3.29}$$

In Equation (3.29), A and ω are assumed as the amplitude and the frequency parameter of the signal, respectively, and b represents the DC bias of the signal. In addition, the shift phase of the two arms trajectories signal is half of the period of each signal, hence by producing the trajectory of one arm the other arm's trajectory can be calculated. Since legs and arms have the same frequency, ω_{arm} can be considered equal to ω legs. According to the fact that the angle of arms are also zero at the start of walking, b factor is assumed as 0. Therefore, arm angular trajectory generator is reduced to the equation (3.30).

$$F(t) = A\,sin(w_{arm} t), w_{arm} = w_{leg} \tag{3.30}$$

In this arm movement generator based on truncated Fourier series, only the value of the parameter A should be obtained in order to generate an arm movement.

3.4.4 Summary

TFS is reported to be a suitable gait representation approach that can generate proper angular trajectories for controlling biped locomotion. It does not require inverse kinematics and also the relation between parameters and generated trajectories is clear. For example stable gaits with different step lengths and stride frequencies were generated by changing the value of only one parameter. Using reduced models for generating balanced gait like TFS, lead to reduce the search space and to generate human-like biped locomotion using machine learning and gait optimization.

All of previous TFS approaches presented in this section were focused on producing walking movements in sagittal plane. We present a new model of hip angular trajectory generator based on TFS which can produce the leg's movement on the coronal plane (Y direction) [16]. This improvement is explained in Chapter 5.

TFS does not have clear design methodology for keeping robustness in the presence of environment changes, disturbance and noise, which exist in real world. Indeed, one of the main issues of the presented model free approaches, such as TFS, sine-based and bio-CPG approaches, is to find ways to relate gait generator parameters to output parameters such as ZMP, and CoP measurements. Presenting an approach where gait designers can also attain additional balance control objectives, seems to become an important future research topic in the area of model free approaches.

Another issue of using TFS approaches is that the methods presented in this section can only generate walking in the forward direction. Generating other walk directions such as side, diagonal, and turn is crucial in humanoid robot locomotion to perform their tasks. We try to generate another walk direction such as turn-in-place by using TFS approach [17][19]. Further discussion about the TFS approaches can be found in the conclusion of Chapter 5, in which the whole model free gait generator is developed based on the TFS approach.

Chapter 4

Gait Optimization and Humanoids Simulation

All biped locomotion approaches explained in Chapter 3 have some parameters in their modeling to generate walking patterns, for which proper values must be found. Robots' dynamic stabilization, walking speed and energy consumption are directly influenced by walking patterns or gaits. Therefore, gait optimization has an important role for legged robot's locomotion. Gait optimization tries to determine the optimal walking characteristics based on position, velocity and acceleration of each Degree of Freedom (DOF).

In this chapter, gait optimization techniques and humanoid simulators are briefly reviewed. In the gait optimization part, we try to address important questions in this field, including how the gait optimization will be performed and which optimization algorithms have already been used in this field. Generally, optimization techniques used for optimizing gaits are organized in two subgroups: local search optimization techniques and population based optimization algorithms. Local search optimization techniques will be discussed in section 4.1, including three optimization techniques: Hill Climbing, Simulated Annealing and Tabu Search. Applications of presented methods will also be reviewed.

Three well-known population based optimization techniques will be introduced in section 4.2 including Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Covariance Matrix Adaptation Evolution Strategy (CMA-ES). Their application on the gait optimization will also be addressed.

Since most of optimization procedures are performed in the simulation environment, a humanoid simulator also has an important role in the gait optimization task. In section 4.3, some humanoid simulators will be reviewed. In section 4.4, the problem of the gap between simulation and reality will also be discussed. Some techniques to bridge this gap will also be reviewed. In section 4.5, conclusion of the topic discussed in this chapter will be given. Also,

some problems that have not yet been solved and can show perspectives for future research in the area of gait optimization will be addressed.

4.1 Local Search Optimization Techniques

The major distinction between local search optimization techniques and population based optimization techniques is that the latter, instead of maintaining a single solution in memory maintain a population of possible solutions. Due to this characteristic and to its simplicity, local search techniques regularly have less search power than population based optimization. As an advantage, they need fewer evaluations, in each iteration, than population based optimization and thus they can converge to a solution in a faster manner. Three main local search optimization techniques will be discussed in this section: Hill Climbing (HC), Simulated Annealing (SA) and Tabu Search (TS).

Local Search techniques are used in gait optimization frequently. As an instance, HC was used to optimize a performance of the walk generated by a ZMP preview control approach [113]. A kick skill was also developed with the use of an optimization algorithm similar to HC [114]. In another Study, both Tabu search and HC were used to optimize a kick skill, where it was reported that Tabu search worked better than HC [115].

Asta et al. used SA to optimize a gait generated by a partial Fourier series approach [100] then compared the performance of the SA with GA in the same gait optimization task [116]. It was reported that SA performs better than GA in the simulation. However, this result is not completely valid, since it is was not tested on a real robot. In order to validate the performance of the walk in simulation, it is hard to discover the amount of environmental noise that exists in the real world and that may significantly affect the performance of the simulated walking.

Since local search techniques often required a low number of iterations to converge to a local optimal solution, they may be used directly on a real robot. For example, we used hill climbing to optimize a turn skill directly on a NAO humanoid robot. This approach will be explained in Chapter 5. The rest of this section contains the description of local search techniques including Hill Climbing, Simulated Annealing and Tabu Search.

4.1.1 Hill Climbing

The Hill Climbing (HC) algorithm is a simple local search technique. First set of solutions are generated randomly, and then attempts to find a better solution by analysing the neighbour alternatives that are obtained by making minor changes to the initial solution. It finishes when reaches a state that no neighbour has a better value.

In HC algorithm, the solution are selected randomly among the set of best successors, if there is more than one. HC often converges quickly to the final solution since it simply can improve bad state to a better one. Unfortunately, in many cases, hill climbing cannot achieve good results, especially when there are some local optimum that are worse than the global one, or there is a plateau, which is a flat area of the state-space landscape. It can be a flat local maximum, from which no better state, or a shoulder, from which progress is possible. A hill-climbing search might get lost on the plateau [117]. Hill climbing is a suitable approach to find a local optimum since finding the global optimum cannot be guaranteed. The flowchart of the hill climbing is shown in Figure 4.1.

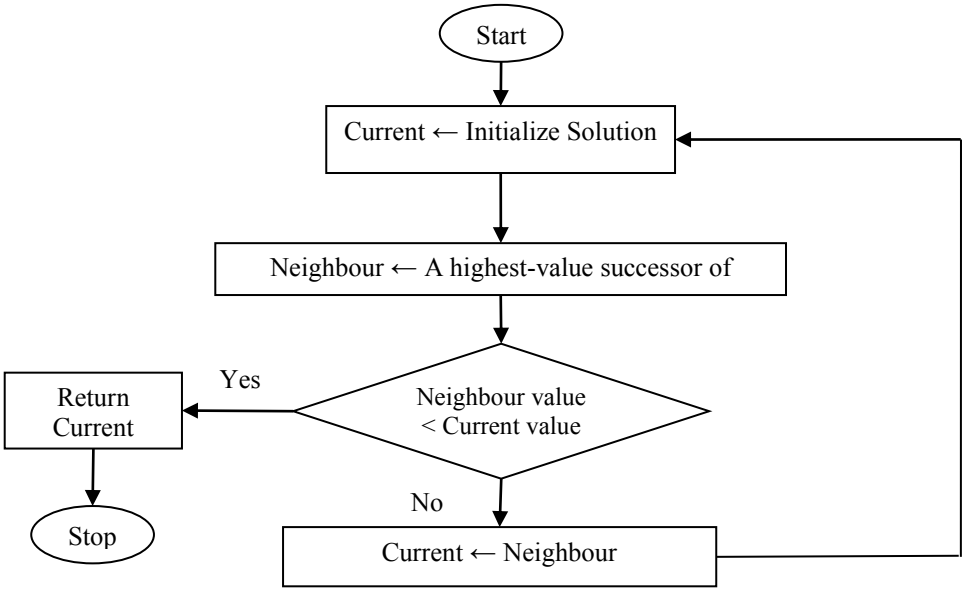

Figure 4.1: Flowchart of Hill Climbing

4.1.2 Simulated Annealing

Simulated Annealing (SA) is a stochastic meta-heuristics local search method. SA is designed to avoid trapping in local, non-global optimums, in its searching algorithm to find global optimums [118]. Simulated Annealing is motivated by the process to temper or harden metals and glass in metallurgy studies. In this procedure, the temperature of material reaches to a high such that the material melts and the atoms can move freely, then the temperature is decreased slowly, in this case, the atoms can move enough to begin adopting the most stable orientation. If the material is cooled slowly enough, the atoms are able to achieve the most stable orientation.

Simulated Annealing algorithm is similar to HC algorithm, however in order to choose the next state, instead of choosing the best state, SA picks a random state. In the case that the random state improves the state and contains the better value, it will be accepted. If not, SA accepts the move with a probability. This probability and the iteration number have an inverse relation, meaning that the probability decreases as the iteration evolved. This relation is modeled by

"temperature" *T* decreasing. In the first iterations of the SA algorithm, the temperature is high, and over time, the temperature will be decreased. Thus, worse states are more likely to be accepted at the first iterations, but this probability will be declined, as iterations will be passed. The Simulated Annealing flowchart is shown in Figure 4.2.

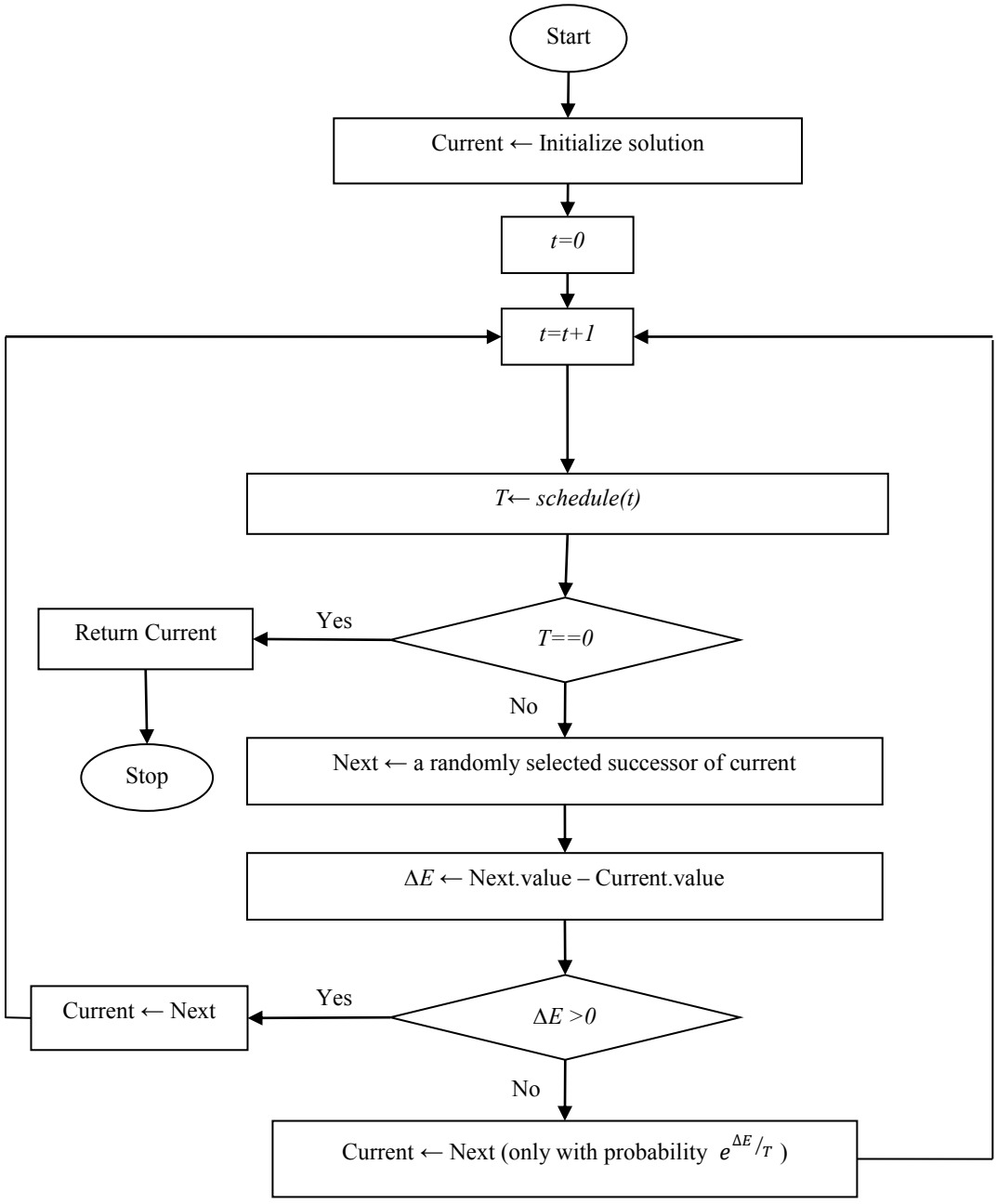

Figure 4.2: Flowchart of Simulated Annealing

4.1.3 Tabu Search

Tabu Search (TS) is designed based on the HC method, which is proposed by Fred Glover in 1986. This algorithm keeps a list of the *k* last visited states to avoid repeating a previous state [119]. This approach prefers to visit worse states rather than repeating them. It also improve

efficiency by avoiding cycles, this improvement can allow to escape from a local optimum, since it can accept worse states [119]. The flowchart of this method is presented in Figure 4.3.

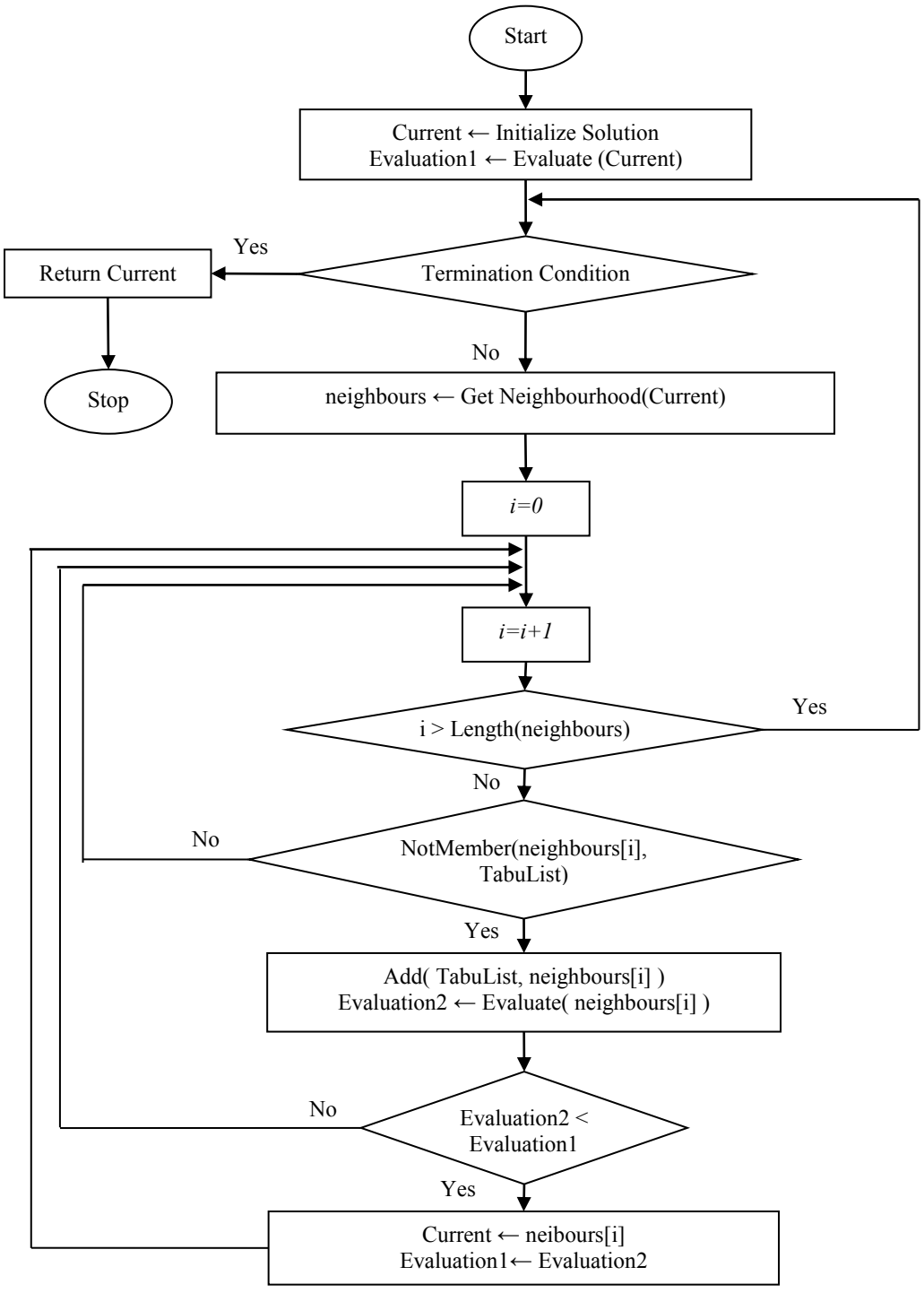

Figure 4.3: Flowchart of Tabu Search

4.2 Population based Optimization Algorithms

Humanoid walking that integrates mobility and stability is a very crucial task for biped robots, since their system of locomotion has multiple DOFs and its dynamics equation is a high order

nonlinear system. In order to model a biped locomotion system especially using model-free approaches, a large number of parameters are considered and their proper values should be found. For example, in one study, 54 parameters are considered for a walk gait of the Sony AIBO robot [120]. The approach presented in 3.4namely GAOFSF [48] needs 8 parameters for generating walking angular trajectories in order to produce walking pattern in sagittal plane. In another study used GAOFSF, 9 parameters are needed to produce all walking movements including arms and legs movements [105].

Gait generation regularly has several purposes, and some of these purposes may conflict with each other, for instance speed and stability. Therefore, it can be represented as a multi-constrained and multi-objective optimization problem.

The dynamic equations of humanoid biped locomotion are high-order highly-coupled and nonlinear [121]. Gait optimization for humanoids needs to find values a set of parameters in a highly irregular and multi-dimensional space. As a result, the optimization methods based on standard gradient search are not suitable to be used in a gait optimization task.

Population based optimization techniques seem to be a suitable solution for gait optimization task. The reason for this is that it has significant global search ability and reduces the danger of becoming trapped in local minima.

Population based optimization also has capability to be parallelized without any problems. In many proposed gait optimization procedures, a robot must test each solution that is calculated by an optimizer. Therefore, the execution of gait optimization may be very time-consuming. Using parallel implementation of gait optimization is one of the advantages of population-based method to improve the rate of convergence.

Population based optimization techniques on the basis of Evolutionary Computation (EC) are repeatedly reported suitable with the gait optimization of legged robots [122], since real world has inherent noise and it is hard to achieve an accurate dynamic model of humanoids [122]. Evolutionary Computation techniques provide robust solution in the presence of environmental noise. Gait optimization based on EC is a mixture of population based optimization methods and EC techniques. Up to now, many EC techniques have been investigated to solve gait optimization problems. The gaits most often studied include gaits of biped [123], quadruped and hexapod robots [124]. They are used to perform movement such as walking, running, negotiating slope surfaces, and going up and down stairs [125].

Genetic Algorithms (GA) and Covariance Matrix Adaptation Evolution Strategy (CMA-ES) are kinds of Evolutionary Computation (EC), which mostly use as the optimization techniques in the gait optimization task. In section 4.2.1, GA and its application on the gait optimization will be described; CMA-ES will be described in section 4.2.3. In section 4.2.2, Particle Swarm

Optimization (PSO) as a well-known population based optimization technique and its application in gait optimization will be introduced briefly.

4.2.1 Genetic Algorithm

Genetic Algorithm (GA) is a stochastic searching procedure inspired by the natural evolution. It is based on the mechanics of natural selection and genetics [126]. In 1975, John Holland first developed Genetic Algorithms techniques based on two theories: exploration and exploitation. It means that GA first tries to expand the search space to find the most appealing parts and after that, it tries to take advantage of those regions to find the extreme more accurately. It should be mentioned that finding global optimum is not guaranteed by this algorithm. A GA allows a population composed of many individuals to evolve under certain selection rules to a state that maximizes the "fitness" or minimizes the cost function [126]. In this case, an individual (also called chromosome) is a set of parameters, known as genes, and represents a possible solution to the optimization problem.

The algorithm starts by creating randomly distributed initial populations of solutions (chromosomes). They contain the values of all parameters, which are needed to be evolved. The design variables have an important role, when treated as genes and combined in an array; building a chromosome. Multiple encoding methods of chromosome exist, including the gray code representation, real number coding, but the most often used encoding is the real coded method. This is due to difficulties associated with binary representation when dealing with a continuous search space with large dimension [127].

Each chromosome will be evaluated by the objective function to attain a score for each of them, which is called fitness. Fitness function has a critical role in GA and is used to judge the goodness of a solution represented by a chromosome. In gait optimization, fitness function are calculated based on the walk performances, such as speed or stability of walking, or energy consumed by a biped.

GA starts the evolution by applying several operators to the population, generating a new one. Along the algorithm, a sequence of new populations is generated. The simplest form of GA involves three types of operators: selection, crossover, and mutation.

Selection is the operator that selects chromosomes in the population for reproduction, according to a given probability. The chromosomes that fit the objective function more will be more likely selected to reproduce the next generation. Elitism method is also a selection method, which determines the number of the best individuals at each generation to survive in the next generation. Such individuals can be lost if they are not elected to reproduce or if they are erased by crossover or mutation. It was reported that this technique can achieve better results [126].

Crossover operation randomly chooses a locus and exchanges the subsequence before and after that locus between two chromosomes to create two offspring (child). For example, the strings 10000100 and 11111111 could be crossed over after the third locus in each to produce the two offspring 10011111 and 11100100. The crossover operator roughly mimics biological recombination between two single-chromosome organisms. Another proposed crossover functions is a scattered function. This function creates a random binary vector equal to the length of chromosome. Then the genes where the produced vector is a 1 from the first parent and the genes where the vector is a 0 from the second parent have been selected, then by combining these genes the child will be formed.

Mutation operator randomly flips some of the genes in a chromosome. For example, the string 00000100 might be mutated in its second position to yield 01000100. In this example, mutation can occur at each bit position in a string with some probability, usually very small, e.g. 0.001 [126]. For mutation operation that is occurred in a chromosome with real number variable, uniform function is also widely used with the probability of occurring as the mutation rate lower than 0.1, a random number in a range of upper and lower bound is selected uniformly and then the existing number in the fraction is replaced by this random number.

Adaptability is also an approach for GA in order to have a better evolutionary optimization by balancing local and global exploitation and exploration. In this case, optimization can modify the probabilities of mutation and crossover adaptively [128]. All of presented functions are iterated and repeated (excluding the initialization phase) until an ending criterion such as desirable fitness or a maximum number of iterations is met. Figure 4.4, shows the flowchart of a GA procedure. The mutation function receives the parameter P_m which is the probability of occurring a mutation in a chromosome.

Some of the advantages of using GA are as follows:

- Optimizing continuous or discrete variables;
- Not requiring derivative information;
- Simultaneously searching from a wide sampling of the cost surface;
- Dealing with a large number of variables;
- Well suited for parallel optimization;
- Optimizing variables with extremely complex cost surfaces. It can jump out of a local minimum;
- Providing a list of optimum variables, not just a single solution.

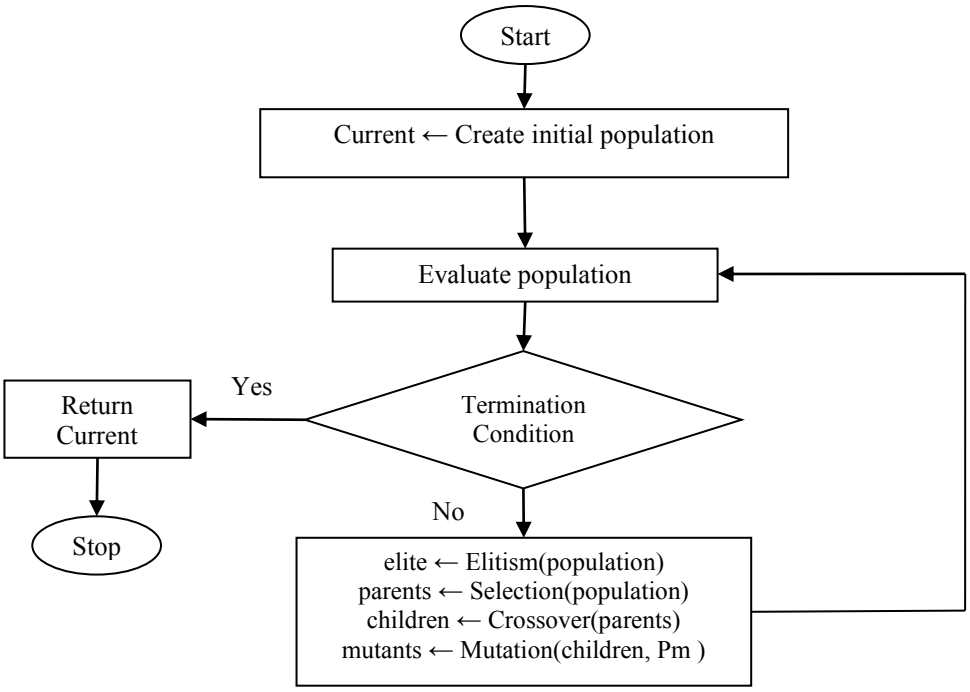

Figure 4.4: Flow chart of Genetic Algorithm

Genetic Algorithm (GA) is suitable for the problems of gait optimization [104] [124]. For example GA is used to find the best parameters to generate angular trajectories for bipedal locomotion [104]. The unambiguous fitness sharing mechanism has also been implemented to avoid untimely convergence to suboptimal extremes[124].

Figure 4.5 shows a biped locomotion which is obtained from a gait optimization in a simulation environment. Gait optimization was done by genetic algorithm, in order to lead the robot to learn how to walk straight.

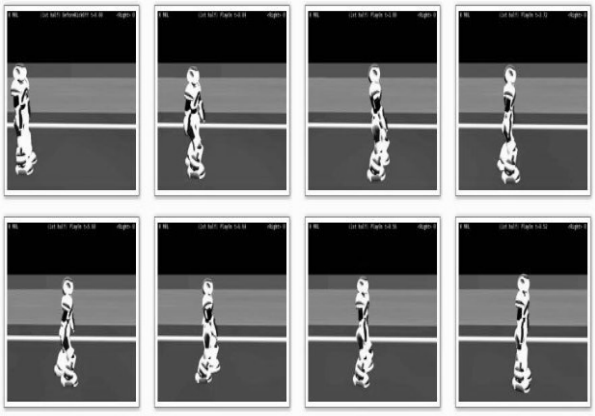

Figure 4.5: Biped Locomotion by using GA [104]

As mentioned before GA is categorized in a kind of Evolutionary Computation (EC). In order to describe how a gait optimization can be designed by an evolutionary algorithm, a general

block diagram of evolutionary computation based gait optimization is given in Figure 4.6. Evolutionary computation algorithm and walking algorithm for the robot are shown on the left and right sides of the figure, respectively.

Parameters to be optimized in a gait model are coded as a chromosome. EC first initializes the optimization algorithm by population of possible solutions randomly, after that, each solution is sent to the robot for testing. Result of a solution will also be calculated based on walk's characteristics, constraints and objective function. EC techniques based on its algorithm try to optimize the value of each parameter and find the best solution. This loop will continue until a sufficient criterion is met (usually a desirable fitness or a maximum number of iterations).

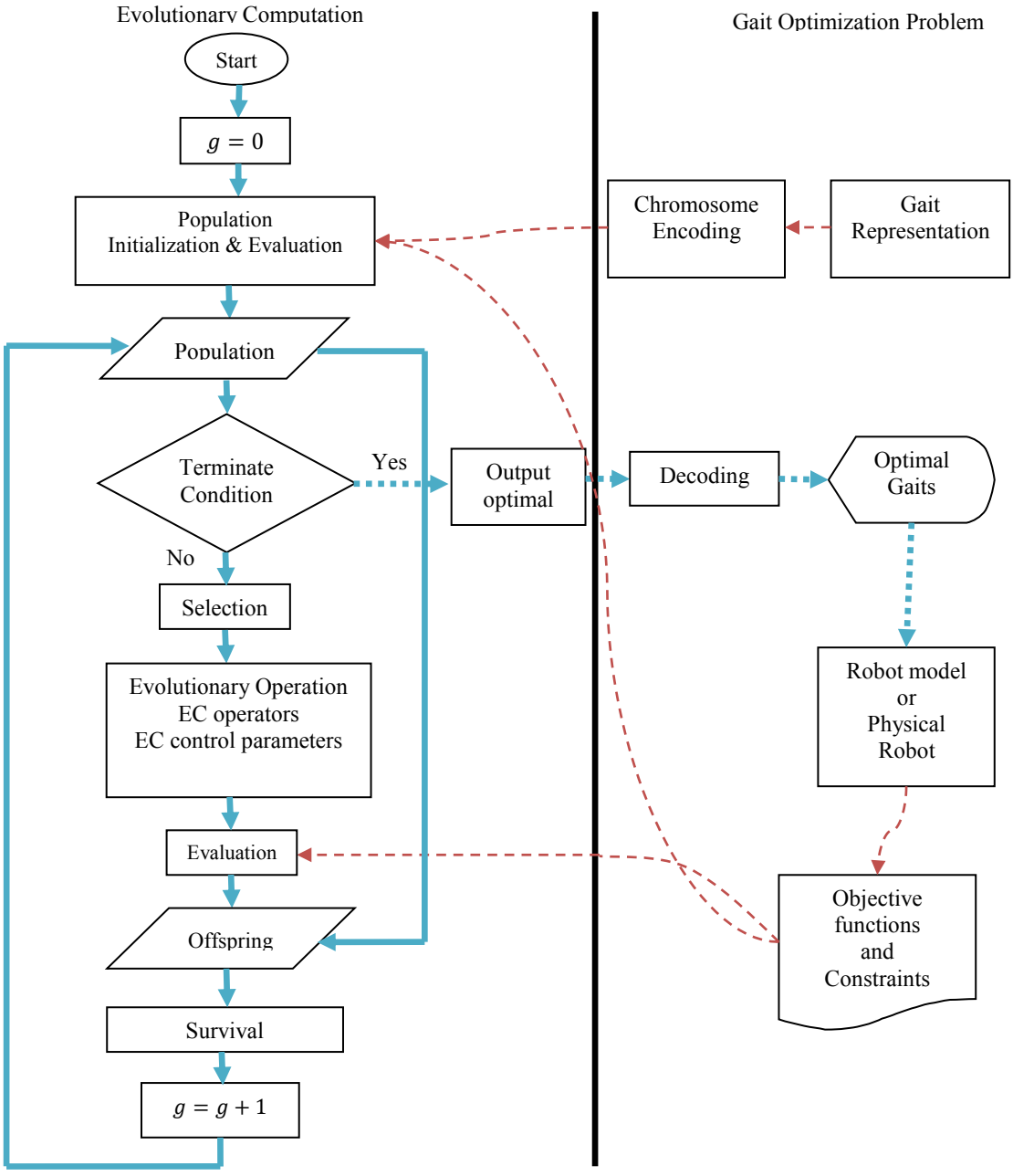

Figure 4.6: Gait optimization diagram based on evolutionary computation [122]

4.2.2 Particle Swarm Optimization

Particle Swarm Optimization PSO [129] is a biologically inspired population based optimization approach. In 2009, it was used for the gait optimization, and it was reported that PSO can achieve better results than GA [103].

Particle Swarm Optimization (PSO) was invented by in the mid 1990s while attempting to simulate the choreographed, graceful motion of swarms of birds as part of a socio-cognitive study investigating the notion of "collective intelligence" in biological populations. In PSO, a set of randomly generated solutions (initial swarm) propagates in the design space towards the optimal solution over a number of iterations (moves) based on large amount of information about the design space that is assimilated and shared by all members of the swarm. PSO is inspired by the ability of flocks of birds, schools of fish, and herds of animals to adapt to their environment, find rich sources of food, and avoid predators by implementing an "information sharing" approaches. Examples of research work on PSO applied for gait optimization can be found in [130],[131]. The PSO algorithm consists of three steps; Initializing particle's positions and velocities, velocity update, and position update.

Initializing Particles' Positions and Velocities

A particle refers to a point in the designed space that changes its position from one move (iteration) to another based on velocity updates. As PSO is a population-based optimization algorithm, each particle is an individual and the swarm is composed of particles. The relationship between swarm and particles in PSO is similar to the relationship between population and chromosomes in GA. The swarm size will be denoted by *N*.

Using the upper and the lower bounds on the design variables values, $\mathrm{X_{min}}$ and $\mathrm{X_{max}}$, the positions, $\mathrm{X_k^i}$, and velocities, $\mathrm{V_k^i}$, of the initial swarm of particles can be first randomly generated. The positions and velocities are given in a vector format where the superscript and subscript denoting i^{th} the particle at time *k*. "*Rand*" is a uniformly distributed random variable that can take any value between 0 and 1. This initialization process allows the swarm particles to be generated randomly across the design space. Equations (4.1) and (4.2) are used to initialize particles, in which $\Delta \mathrm{t}$ is the constant time increment.

$$\mathrm{X_0^i} = \mathrm{X_{min}} + \mathrm{Rand}\ (\mathrm{X_{max}} - \mathrm{X_{min}}) \tag{4.1}$$

$$\mathrm{V_0^i} = \frac{\mathrm{X_{min}} + \mathrm{Rand}(\mathrm{X_{max}} - \mathrm{X_{min}})}{\Delta \mathrm{t}} = \frac{\mathrm{Position}}{\mathrm{time}} \tag{4.2}$$

Updating the Velocities

By using the fitness values that are functions of the particles current positions in the design space at time *k*, the velocities of all particles at time $k + 1$ are updated**.** The fitness function

value of a particle not only determines which particle has the best global value in the current swarm,P_k^g, but also determines the best position of each particle over time, P^i, in the current and all previous moves. The velocity update formula uses these two pieces of information for each particle in the swarm along with the effect of current motion, V_k^i, to provide a search direction, V_{k+1}^i for the next iteration and to ensure good coverage of the design space and avoid entrapment in local optima.

The velocity update formula includes some random parameters, *rand*, represented by the uniformly distributed variables. The three values that affect the new search direction, namely, current motion, particle own memory, and swarm influence, are incorporated via a sumsmation approach as shown in Equation (4.3) with three weight factors, namely, inertia factor, w, self-confidence factor, C_1, and swarm confidence factor, C_2. The constraint in equation (4.4) formed by $\mathrm{V_{max}}$, a maximum velocity, is specified to clamp the excessive accelerations.

$$V_{k+1}^i = w\, V_k^i + C_1\, Rand\, \frac{(P^i - X_k^i)}{\Delta t} + C_2\, Rand\, \frac{\left(P_k^g - X_k^i\right)}{\Delta t} \tag{4.3}$$

V_{k+1}^i : Velocity of Particle i at time $k+1$

V_k^i: Current Motion

$Rand\ \frac{(P^i - X_k^i)}{\Delta t}$: Particle Memory Influence

$Rand\ \frac{\left(P_k^g - X_k^i\right)}{\Delta t}$: Swarm Influence

$$\begin{aligned} &\text{if } \left(\mathrm{V_{k+1}^i} > \mathrm{V_{max}}\right);\ \mathrm{V_{k+1}^i} = \mathrm{V_{max}} \\ &\text{if } \left(\mathrm{V_{k+1}^i} < -\mathrm{V_{max}}\right);\ \mathrm{V_{k+1}^i} = -\mathrm{V_{max}} \end{aligned} \tag{4.4}$$

The inertia weight w controls how much of the previous velocity should be retained from the previous step. A larger inertia weight facilitates a global search, while a smaller inertia weight facilitates a local search. A balance can be achieved between global and local exploration to speed up search results using a dynamically adjustable inertia weight formulation. Introducing a linearly decreasing inertia weight into the original PSO significantly improves its performance through the parameter study of inertia weight [132]. The linear distribution of the inertia as an adaptability approach is expressed as follows [69]:

$$w = \mathrm{w_{max}} - \frac{\mathrm{w_{max}} - \mathrm{w_{min}}}{\mathrm{k_{max}}} \times k \tag{4.5}$$

Where $\mathrm{w_{max}}$ and $\mathrm{w_{min}}$ are the initial and final values of weighting coefficient, respectively, and $\mathrm{k_{max}}$ and k are the maximum iteration number and iteration counter respectively.

Instead of (4.5), Utilizing a nonlinear decreasing inertia weight as a dynamic inertia weight significantly improves its performance through the parameter study of inertia weight [132]. This nonlinear distribution of inertia weight is expressed as follow:

$$w = \mathrm{w}_{\mathrm{init}} * U^{-k} \tag{4.6}$$

$\mathrm{w}_{\mathrm{init}}$ is the initial inertia weight value selected in the range [0, 1] and U is a constant value in the range [1.0001, 1.005], and k is the iteration number.

Updating the Position

Position update is the final step of a PSO-iteration and is perform using the current particle position and its own updated velocity vector shown in the Equation (4.7).

$$X_{k+1}^{i} = X_{k}^{i} + V_{k+1}^{i}\Delta t \tag{4.7}$$

Updating the position is an important process in PSO algorithm to become convergent. When an optimization problem has constraints, feasible solutions that satisfy the constraints should be found. Since a solution vector because of its exploration velocity, may dissatisfy a constraint, there is a common methodology to generate feasible solutions, in which the feasibility of a generated solution in each step is checked. As a result, if a solution vector does not satisfy a constraint, the related vector solution will be punished by a big penalty on its fitness.

Figure 4.7 shows the flowchart of the presented PSO algorithm. We use PSO, to optimize walking angular trajectories produced by a model-free approach; this approach will be described in Chapter 5.

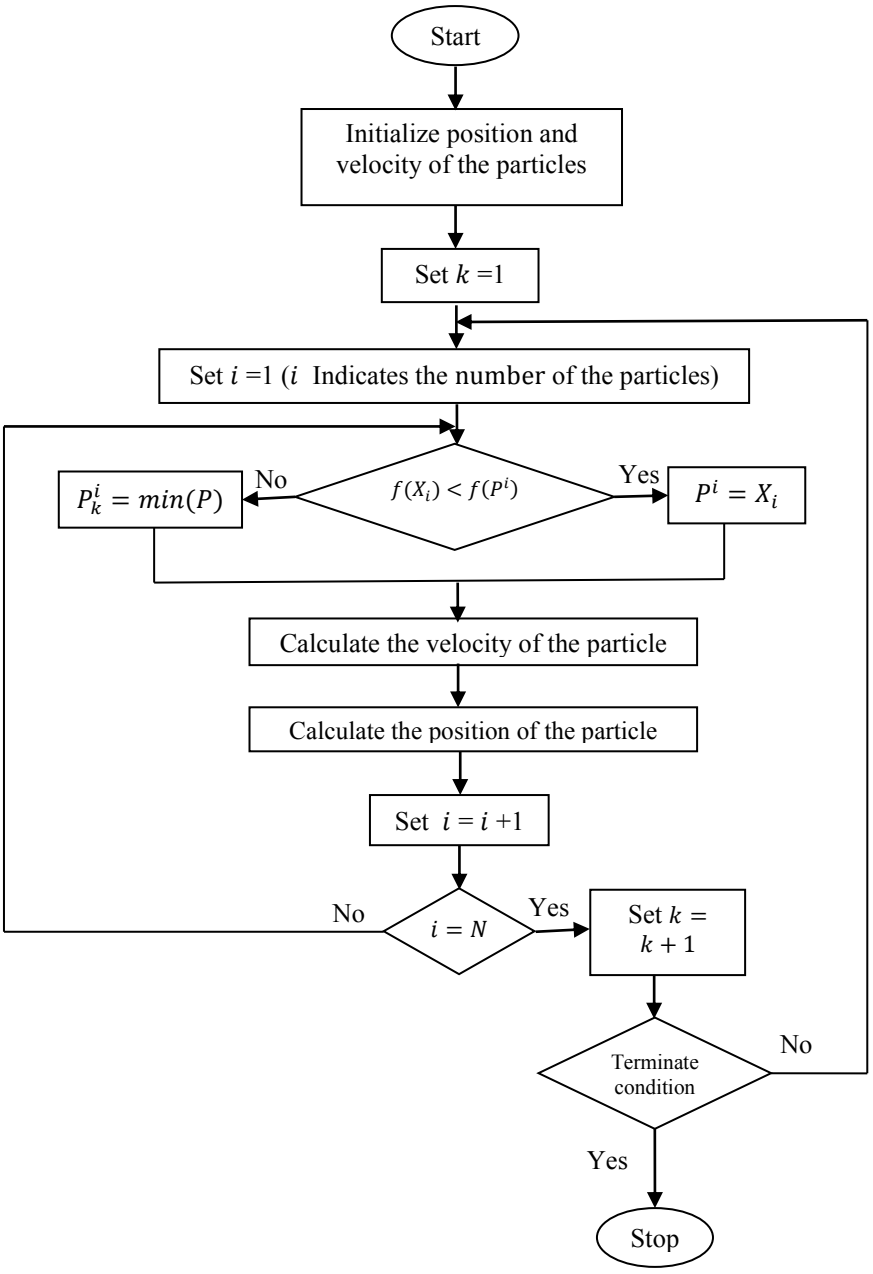

Figure 4.7: Flowchart of PSO

4.2.3 Covariance Matrix Adaptation Evolution Strategy

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [133] is a stochastic optimization algorithm that can be used for gait optimization scenario. CMA-ES is a population-based stochastic, derivative-free method, which can be used in black-box optimization problems or it can be used as direct policy search in reinforcement learning methods. It has been successfully applied previously on gait optimization scenarios [134] [135]. It is also reported that CMA-ES can achieve better results and faster convergence in gait optimization scenarios compared to other famous population based optimization techniques such as particle swarm optimization (PSO) and Genetic Algorithm (GA) [135] [101].

CMA-ES generates a set of solutions, as the population, sampled from a multivariate Gaussian distribution. After generating the population, CMA-ES evaluates each solution with respect to a fitness measure. After evaluating all generated solutions in the population, Multivariate Gaussian distribution attributes are updated based on the solution with highest measured fitness. The Gaussian distribution mean is obtained by using weighted average of the solutions with the highest fitness, the solution with higher fitness has a higher weights. The covariance matrix of the distribution is updated such that the direction of the next generated candidates goes toward previously successful solutions. The CMA-ES Flowchart is given in Figure 4.8.

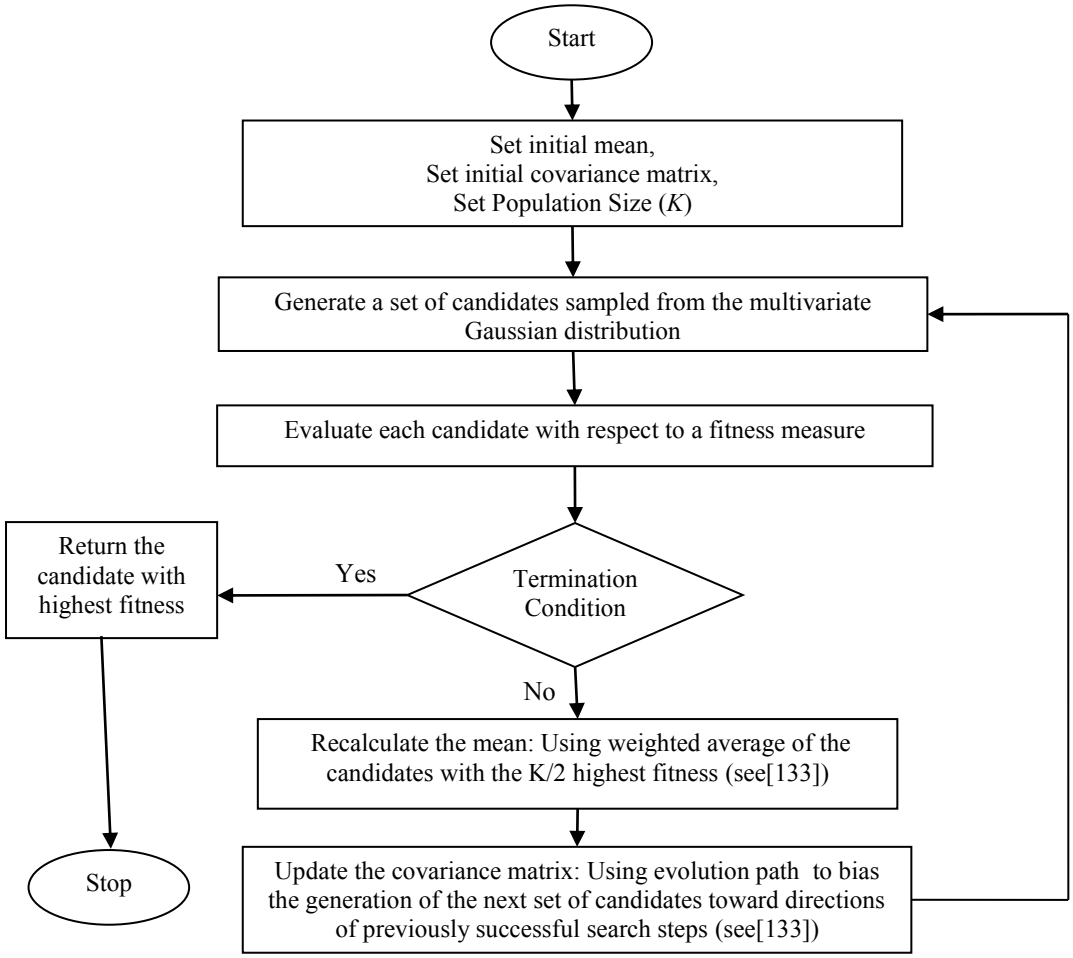

Figure 4.8: Flowchart of CMA-ES

4.3 Humanoid Simulators

The idea behind a humanoid simulator is to develop a virtual agent capable of thinking and acting. Therefore, acquired knowledge from simulation can be transferred to real robots. Simulation is an easier way to develop biped locomotion methodologies. It makes performing gait optimization and learning phase easier and more efficient [136]. It is also not reasonable to

test the algorithm directly on the real humanoid robot since learning needs lots of iterations and humanoids always are very expensive and hard to troubleshoot in the case of falling or breaking. Therefore, humanoid simulators must emulate the real world in the best possible way. To make this feasible, it is necessary to construct accurate and reliable models of the real robots. On the other hand, physical simulators always contain some simplifications when it intended to model the real world, meaning the result of simulation and reality is usually not the same [137]. This problem is called the reality gap in the field of evolutionary robotics.

Researchers always try to find a simulator which has the least differences between its result and reality [138]. In this case of study, some researchers try to compare different simulators and find the differences between simulators in order to identify exact causes of differences between them. It can also lead them to obtain the best model of a simulator that is closer to the reality. Performance of humanoid's simulation depends on many factors related to modeling the physical environment and each simulator uses its specific architecture such as physics engine and methods for discretizing time. A clear approach to compare efficiency of simulating humanoid does not exist yet.

In 2009, Kalyanakrishnan et al. compared three humanoid robotics platforms [139]. They compared the simulators by analyzing results of robots walking with the same robots model and same approach of walking. For example, they compared the speed of walking in simulators with the speed of walking in reality, so as to conclude which simulator is closer to the real world.

In this section, in order to be familiar with humanoid simulator structure, we will try to review different humanoid simulators, including SimTwo [140],Simspark (RoboCup 3D league Simulator) [15], OpenHRP, Webots and Gazebo.

4.3.1 Simspark

The RoboCup 3D league Simulator (Simspark) is a generic simulation platform for physical multi-agent simulations and it is used in the RoboCup 3D simulation league currently. This simulator has been developed by the RoboCup community since 2006. It is designed as a flexible application framework and intends to be a generic simulator, capable of simulating anything, from the launch of a projectile for academic purposes to a big soccer game for scientific research purposes. The framework facilitates exchanging single modules and extending the simulator [141]. The Simulator consists of two important parts: Server, and Monitor.

The server is responsible to handle connections from the agents; to receive and to process messages, send reply messages to the agents. The server architecture is illustrated in Figure 4.9.

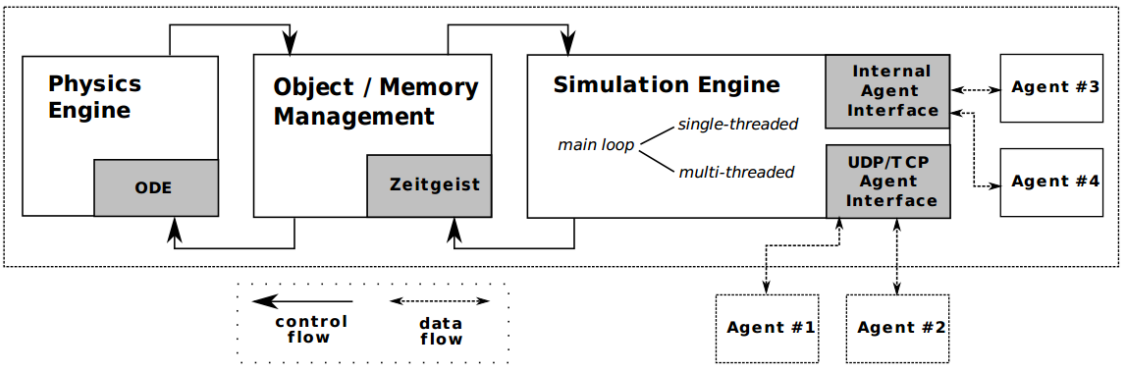

Figure 4.9: RoboCup Simulator Server Architecture [141]

The simulator uses the Open Dynamics Engine (ODE) to simulate the physical environment. ODE is a physical simulation engine, which allows simulating the system's dynamics and the physical properties of the simulated objects. It provides advanced joint types and integrated collision detection with friction. ODE is particularly useful for simulating objects in virtual reality environments. It is cross-platform and provides a user-friendly C/C++ Application Programming Interface (API).

RoboCup simulator is very well-known because it can be seen as a multi-agent environment enabling different types of experiments. Several agents can connect to the server simultaneously. Its object and memory management is based on Zeitgeist [141]. It is a framework for handling data objects and functional components of a system in a uniform way, which strictly follows the object-oriented paradigm using C++ programming language.

The core of the simulator is a simulation server. It receives the messages with actions from the agents, performs the simulation operations and sends a reply message back to the agent with the environment information. The simulation engine acts as a server, handling the messages of the agents and replying on to other messages. In each cycle of the simulation, agents send a message to the server containing information about their effectors (e.g. joints). The message from the server to the agent contains temporal information and specific information from the application domain (which is soccer). This information includes game state (play mode, time and current result), and information of the sensors of the robot (e.g. joints, gyroscopes, foot sensors, vision information). The messages are constructed using a LISP-like format.

The monitor provides a simple graphical interface that allows the user to watch a simulation. Simulations may be watched in real-time, but it is also possible to play simulation log files offline, for post processing assessment. In the particular case of humanoid soccer simulation, it provides additional information such as the team names and the game time, play mode and results. Several shortcut keys can be used to change camera views, to drop the ball, and carry out other useful operations.

4.3.2 SimTwo

In 2006, SimTwo [140] was developed by Dr. Paulo Costa in the University of Porto. It focused on preparing a simulated environment to test and develop mobile robot approaches and applications.

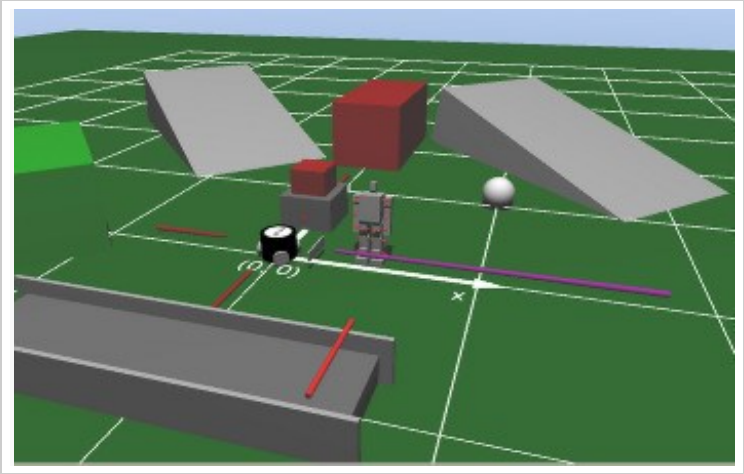

Figure 4.10: Snapshot from SimTwo environment

The SimTwo system is a realistic simulation where anyone can implement various types of mobile robots with different configurations including the different body mass and shape and any types of joints, which can be described with a mixture of classic joints and wheels. Therefore, in addition to simulate wheeled robots, humanoid robots can also be simulated. Recently, the model of NAO robots was also added to this simulator. In Figure 4.10, the environment of SimTwo is shown.

The realism of the dynamics implemented in SimTwo is achieved by dividing a robot into a system of rigid bodies and electric motors. The "mechanics" associated with bodies is numerically simulated considering their physical shape, and mass moments of inertia, friction and elasticity of the surfaces. Like in the RoboCup simulator the physical simulation is performed by using ODE, which was described in the previous section. Certain joints are designed as shaft gimbals typically and as pipeline explicitly. Each of them can have its associated drive system and sensors.

The drive system may consist of a DC motor and gearbox. The DC motor model contains several elements such as nonlinear saturation of the applied voltage, current limit and friction. To define the environmental configurations in the simulator SimTwo can design any type of obstacles that may have different shapes and roles such as slops, stairs, cubes and so on. Obstacles and robots can also be moved easily during the simulation by dragging mouse on them.

To define and implement the environmental configuration and robot's model, SimTwo has different tools, which allows developers to import their configuration as XML scripts. In order to show the results of the simulation, a graphical viewer based on OPEN-GL is also implemented. A debugger tool is also developed as a graphical viewer, which can draw any graphs and trajectories. In Figure 4.11 tool-boxes of the SimTwo can be found. The two windows on the right show the XML-based environment as well as the visual viewer and debugger on the left.

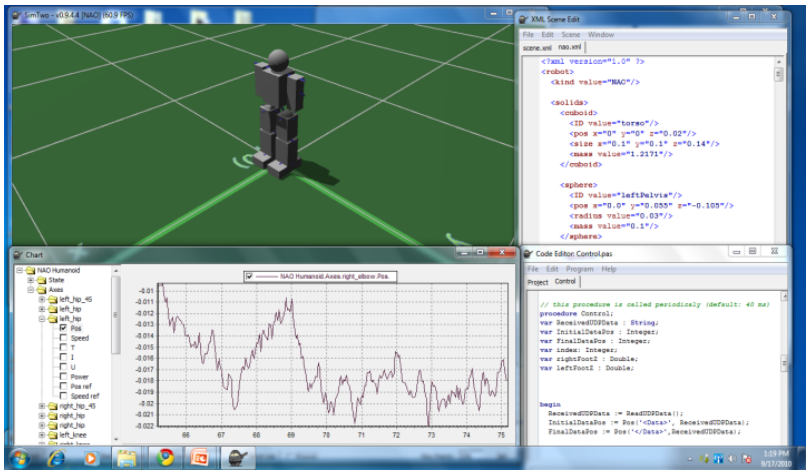

Figure 4.11: View of SimTwo tool boxes

SimTwo has good capabilities to simulate actuators and to define obstacles and environment features, but it does not have adequate architecture to manage several agents to connect to the simulator as a multi-agent environment.

In 2011, we compared RoboCup 3D league Simulator (Simspark) with SimTwo concerning their capabilities to simulate humanoid robots. Analyzing the comparison results led us to conclude that the Simspark is closer to reality than SimTwo in what concerns humanoid robot simulation [142].

4.3.3 OpenHRP

OpenHRP (Open Architecture Humanoid Robotics Platform) is a simulator which is developed to simulate the humanoid robot with its complexities [143]. As a virtual humanoid robot platform, It focuses on simulating the HRP series humanoid robot. In addition to the simulator, a motion control library for HRP robot is also implemented in the OpenHRP platform. It has open-source software that consists of a visualiser, dynamics simulator, motion planners and motion controllers. The software components are communicating with each other using CORBA (Common Object Request Broker Architecture), that is designed to facilitate the communication of systems that are deployed on diverse platforms. The configuration of each component is implemented based on VRML (Virtual Reality Modeling Language).

4.3.4 Webots

Webots is a commercial simulation software that is developed by Cyberbotics Ltd. [144]. It can simulate many models of existing mobile robotics, including humanoid robots and wheeled robots. The provided robot libraries enable transfer of control programs to commercially available real mobile robots such as NAO [145]. It has different components such as physical simulation engine, simulation view and different simulated robot models, which communicate with each other through TCP/IP protocol. The ODE is used in its physical simulation engine.

4.3.5 Gazebo

Gazebo is a three dimensional dynamics simulator, which can simulate multi robots environment with graphical interface [146]. A large variety of sensors and models of existing robots are prepared in the simulator. The Player's client/server model [147] controls the robots and sensors; So that the controllers can be implemented by using a library provided with the simulator. The descriptions of the simulated environment are given in XML format by using predefined robots and elements. It is possible to create and integrate robots as plugins; however this has to be implemented by code-based modeling in C. Gazebo has its own built-in physical engine. In 2012, DARPA (Defense Advanced Research Projects Agency) used Gazebo simulator for the DARPA Robotics Challenge (DRC). DRC aims to develop semi-autonomous humanoid robots that can do complex tasks during a disaster response.

4.4 Bridging the Gap Between Simulation and Reality

Gait optimization must be performed in simulation environment typically, because optimization needs to perform lots of iterations and a real robot does not have this capability. Physical Simulator always contains some simplifications when it wants to model the real world, therefore the result of simulation and reality is usually not the same, however they have many similarity with each other. This problem is called *the reality gap* in the field of evolutionary robotics [137].

This problem is studied by a branch of artificial intelligence concerned with automatic generation of autonomous robots. One of the good approaches to reduce the results of this problem is to integrate evolution in simulation (off-line learning) and in reality (on-line learning) with considering the reality gap. *Staged evolution* is one of the methods to reach this purpose [148] [138]. This method evolves the gait by simulation first so preliminary result for producing the gait is the best one which is achieved in the simulation. Then the simulation results are transferred to the physical robot where the process of evolution is continued on the real robot (but just by searching near to solution) of the previous step.

Another method that has significant result in this field was presented in 2007 [137]. It uses the differences between the behavior fitness obtained in reality versus that of simulations as a feedback and also an objective function for minimizing another optimization scenario. It tries to find the solution which has the least difference between result of simulation and reality; and also best result in simulation.

The flowchart of this approach is explained in Figure 4.12. In step 0, Gaits of Walking based on gait generator model will be produced. In step 1, the best value of gait's parameters in the simulation environment will be found by using evolutionary computation techniques. In step 2, a group of solutions which have good results in the simulation will be selected. In step 3, the selected solutions will be searched and optimized again based on their fitness function.

The fitness function here is calculated based on the difference of the result achieved in simulation and reality, and the goal of the optimization is minimizing this objective function. Each test on the real robot will be executed in step 4. Result of optimization in reality goes to step 2 to search again in simulation, but this time optimization technique tries to find the solution closest to the solutions achieved by step 3 and 4. Finally if the gap between simulation and reality is reduced enough, the optimization will stop.

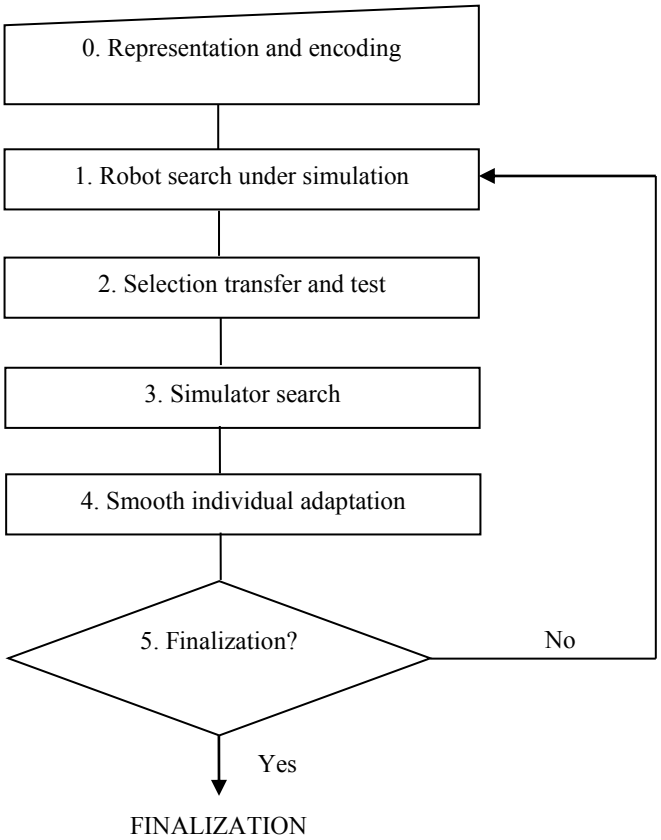

Figure 4.12: Flowchart of the algorithm used for bridging the gap between simulation and reality [74]

4.5 Summary

The most important point that can be concluded from the literature review presented in this Chapter is that, although gait optimization methods can achieve good performance for optimizing gaits in the simulation environment, these methods must be improved more in order to use gait optimization directly on a real humanoid robot efficiently.

One of the most important issues in gait optimization using population based methods is its computational efficiency. Population based optimization methods usually need to generate a population of solutions, and in each iteration this population must be evaluated by its fitness function. Therefore, it is time-consuming to calculate the objective function of all individuals of the population when testing is supposed to be executed on a physical robot. In some cases it may take several days also requiring manual supervision by human during all the experiment time.

Some researchers suggested using innovative operators instead of usual crossover operators, to increase the efficiency of EC methods and reduce the size of the population. They called this operator: *interpolation* and *extrapolation* [148]. Recently, some researchers also reported that by using local search techniques like simulated annealing, gait optimization was performed more efficiently compared to population based approaches [116].

The objective function has an important role in a gait optimization task. The fitness function must be proper along with the nature of the problem, such as robot's platform or the desired performance of the walk. Although complex fitness functions can lead the optimization to a better result, they will decrease the speed of convergence. Thus, constructing suitable objective function needs to have good knowledge and experience and also testing sets of objective functions to know which one is better.

The approach that is used to model the gait is also very important, since it defines the required number of parameters to be optimized. Fewer parameters improve the speed of convergence dramatically. Therefore, studying how to find the best approach with fewer parameters is needed, as mentioned in previous section, up to now, TFS is one of the best gait generator models (as a model-free approach) by using only 6 parameters to produce forward walking angular trajectories in sagittal plane [104].

In ZMP based approaches presented in section 3.2, also there is a possibility to use the optimization techniques. Since the dynamics stability is modeled by the approach itself, these type of approaches usually have less parameters left for optimization. Therefore, gait optimization will be more efficient for them. We will use gait optimizations in both model-free and model-based approaches in Chapter 5 and Chapter 7, respectively.

Another fact that was addressed in this chapter is that gait optimization studies regularly perform in simulation environment. All the humanoid robot simulators presented in this Chapter have similar features, such as full rigid-body dynamics, visualization, robot model formats, and communication between simulators components briefly summarizes the characteristics of these features for the presented humanoid simulators.

Table 4.1: Features of the presented simulators

Simulator	*Robots*	*Physic Engine*	*Configuration*	*Architecture*
Simspark	Sphere, NAO	ODE	Ruby	TCP/IP
SimTwo	Arm, Humanoid	ODE	XML	Monolithic
OpenHRP	HRP series	own	VRML	CORBA
Webots	Vehicle, Humanoids	ODE	VRML	TCP/IP
Gazebo	Vehicle, Humanoids	own	XML	TCP/IP CORBA

In this thesis, we mainly use Simspark as our test-bed. The main reasons to choose this simulator are concerned with the fact that Simspark has a very well developed software architecture with an open source licence which is supported by a maintenance community. Secondly, we have a team to participate in the RoboCup competition and many other teams participating in the soccer robot's competition use Simspark. Therefore, the use of Simspark enables us to compare the performance of our locomotion approaches with other teams in a standard way.

Chapter 5

Truncated Fourier Series Approach Applied on Humanoid Robots

This chapter presents the results achieved by our attempt to improve model-free walking approaches based on Truncated Fourier Series (TFS) on humanoid robots, which led us to publish the following articles:

> **N. Shafii**, L.P. Reis, N. Lau *"Biped Walking using Coronal and Sagittal Movements based on Truncated Fourier Series"*, RoboCup-2010: Robot Soccer World Cup XIV, Lecture Notes in Computer Science, vol. 6556, pp. 324-335, Springer, 2010.

> **N. Shafii**, A. Abdolmaleki, N. Lau, L.P. Reis "*A Robust Closed-loop Gait For Humanoid Clock-Turning*", Proceedings of Workshop on Optimality Principle and Adaptation in Humanoid Robotic Control in conjunction with the 2012 IEEE-RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 27-41, 2012.

> E. Domingues, N. Lau, B. Pimentel, **N. Shafii**, L. P. Reis, A.J.R. Neves *"Humanoid Behaviors: From Simulation to a Real Robot"*, Progress in Artificial Intelligence, Lecture Notes in Computer Science, vol. 7026, pp. 352-364, Springer, 2011.

5.1 Introduction

The previous chapters presented a review on related work and a description of the main methodologies concerning biped locomotion, gait optimization and humanoid simulation. In this chapter, a model-free approach is proposed, in which Truncated Fourier Series (TFS) is used to generate fast forward walking and turn-in-place.

As mentioned in section 3.1, the approach that use Central Pattern Generators (CPGs) or Truncated Fourier Series (TFS) are categorised as model-free approaches. However, in CPG-based approaches, it is difficult to design the relation and feedback pathways of the neural oscillators and to tune the required parameters to achieve the desired walking characteristics manually [149]. TFS based approaches do not have this issue and in this chapter we focus on TFS based approaches.

In 2007, TFS was used to generate walking angular trajectories, in which a ZMP stability indicator was applied to prove that it could generate angular trajectories of a stable forward walk, but it was not implemented on a real robot [48]. In 2009, an optimized gait generator based on TFS was implemented in a simulated humanoid robot to generate forward walk, and TFS parameters were also reduced by 2 dimensions [104]. Shafii et al. extended the basic TFS enabling the generation of arm angular trajectories that provide smooth and robust walking, also a new method was used to refine walking trajectories for reducing the role of inertia and to improve the speed and robustness of the robot [105]. These TFS based approaches enable to produce walking movement only in sagittal plane. In this chapter, TFS based approaches are improved such that they enable to generate the walking movement in joint space in coronal plane and transverse plane. The aim of modeling the coronal movement is to study the role of this movement in generating fast forward walking, and the aim of modeling transverse movement is to enable the TFS based approaches to generate turn-in-place motion.

This chapter is organised as follows. Section 5.2proposes an approach to model angular trajectory of turn-in-place and forward walking. This approach like other model-free approaches has unknown parameters in the modeling of walking movements. These parameters values should be found in learning scenarios. The parameters values of coronal movement are optimized to generate fast forward walking. The parameters values of transverse plane are also learned to generate fast and stable turn-in-place motion. In section 5.4, the learning scenarios for optimizing the TFS parameters in simulation will be presented. As mentioned in section 4.4, the results achieved in simulation cannot be applied on real robot directly, since there is usually a gap between the performance of a robot in simulation environment and in real world. In section 5.5, an approach is presented to modify the learning results obtained using the simulator so that they can be applied on real robot.

The result of the proposed approach and learning scenarios for both simulated and real humanoid robot are presented in section 5.6. Finally, a conclusion that addresses the cons and pros of the proposed TFS based approaches is given in section 5.7.

5.2 Angular Trajectories Modeling

A gait is a cyclic, periodic motion of the joints of a legged robot, which requires the sequencing and coordination of the legs to obtain reliable locomotion [122]. In other words, gait is the temporal and spatial relationship between all the moving parts of a legged robot [122]. Therefore, all gaits angular trajectories are periodic and Truncated Fourier Series (TFS) can be used to model and generate them.

The basic TFS approach was fully explained in section 3.4. The aim of this section is to present TFS based approaches that enable the generation of stable turn-in-place motion and fast forward walking. To generate a forward walk, both sagittal and coronal movement are modeled. Beside movements in these two planes, movement in plane also needs to be modeled in order to generate turn-in-place motion.

In the following subsections, first we will explain how to model angular trajectories of legs and arms joints that move in sagittal and coronal planes to generate forward walking. Then, our approach to model legs angular trajectories for generating turn-in-place will be explained. In this later approach, legs are modeled to move in sagittal, coronal and transverse planes.

5.2.1 Forward Walk Modeling

In this section, a new TFS based approach is proposed to model the legs movement in coronal plane. The aim of the proposed model is to study the role of coronal movement in generating a fast forward walk. In order to generate forward walk, the walking movements are modeled considering both displacements in the sagittal plane and coronal plane.

A) *Modeling of Legs Movement in Sagittal Plane*

The TFS based approaches that were presented in section 3.4 will be used to model walking movements in sagittal plane. There are three DOFs in each leg move in sagittal plane: one at hip, one at ankle and one at knee. In this approach, similar to [150], the feet were kept parallel to the ground by using the ankle joint. This is done in order to avoid collations of swing foot colliding with the ground. In this case, ankle trajectory can be calculated by hip and knee trajectories.

In order to model hip and knee angular trajectories, each of them is divided in two parts; the upper portion and the lower portion. Let us define t_1 and t_2, as a start and end time for the support phase, in which the variation of the knee angle is assumed to be fixed. This duration is also called knee lock phased [104]. During t_1 and t_2, the upper portion is modeled and in the remained time of the walk step the lower portion is modeled. The trajectories for both legs are identical in shape but are shifted in time relative to each other by half of the walking period.

Therefore, by producing the trajectory of one leg the other leg trajectory can be calculated. The TFS for generating each portion of hip and knee trajectories are formulated in equation (5.1).

$$\begin{gathered}\theta_h^+ = A_{hx}.sin(i\omega_h t) + C_{hx}, \omega_h = \frac{2\pi}{T_h} \\ \theta_h^- = B_{hx}.sin(i\omega_h t) + C_{hx}, \omega_h = \frac{2\pi}{T_h} \\ \theta_k^+ = A_k.sin(i\omega_k t) + C_k, \omega_k = \omega_h \\ \theta_k^- == C_k \geq 0\end{gathered} \tag{5.1}$$

In this equation, the plus (+) sign represents the upper portion of walking trajectory and the minus (-) shows the lower portion. A_{hx}, B_{hx}, A_k are constant coefficients for generating trajectories. The h and k subscripts stand for hip and knee, respectively. C_{hx} and C_k are trajectory offsets, and T_h denotes the period of hip trajectory. Considering the fact that all joints in walking motion have equal movement frequency and stride rates are statistically equal [150], the equation $\omega_k = \omega_h = \frac{2\pi}{T_h}$ can be derived. Using the method that was presented in section 3.4.2 and originally discussed in [104], the start and end time of the lock phase can be calculated, therefore the two parameters of t_1 and t_2 are eliminated. In the model presented by (5.1), there are 6 parameters, including C_{hx},C_k, A_{hx}, B_{hx}, A_k and T_h, their values should be learned in order to produce legs movement of a fast and stable forward walk.

B) *Modeling of Arm Movement in Sagittal Plane*

Arm movement has an important role in generating fast forward walking [105]. The approach used for modeling the arm movement in sagittal plane was explained in section 3.4.3, in which the arms trajectory was modeled as sinusoidal signals. Thus, to produce the angular trajectories of arms swinging, it is enough to obtain proper parameters for the equation (5.2).

$$\theta_{arm} = A_{arm}\, sin(\omega_{arm} t), \omega_{arm} = \omega_h \tag{5.2}$$

Where A_{arm} and ω_{arm} are assumed as the amplitude and frequency of the angular trajectory, respectively. The shift phase of the two arm trajectories is half of the period of each signal, so by producing the trajectory of one arm the other arm trajectory can be calculated. Since legs and arms have the same frequency [105], ω_{arm} can be considered equal to ω_h. According to equation (5.2), only the proper value of A_{arm} must be obtained and it will be obtained together with legs trajectory parameters in a learning scenario.

C) *Modeling of Leg Movements in Coronal Plane*

The range of motion in the coronal plane is smaller than what has been seen in the sagittal plane, but it has an important role to keep the balance of walking [151]. The range of coronal

motion depends on the speed of walking, and at higher speeds this range is smaller. Coronal plane movements are periodic motions [152]. Abduction and adduction are terms for movements of limbs relative to the coronal plane. To produce legs motion in coronal plane, we proposed a walking scenario that is shown in Figure 5.1, which illustrates the walking sequence in a walking period and the θ is assumed as the maximum of legs movement. Similar to the leg movement in sagittal plane, feet were kept parallel to the ground. Considering of the fact that just one of each hip joint moves each time, the angle of ankle is equal to the hip angle of the opposite leg.

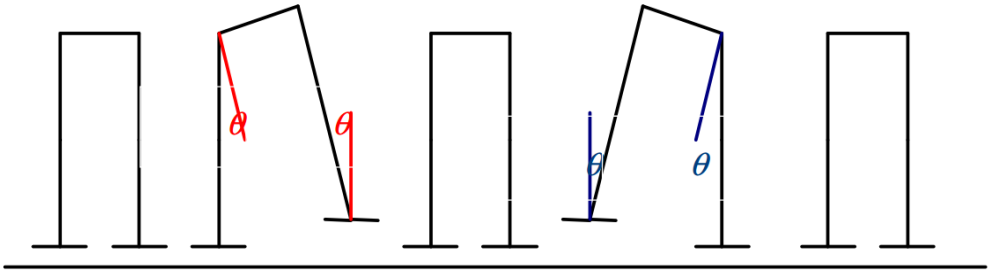

Figure 5.1: Coronal Plane review of walking sequence

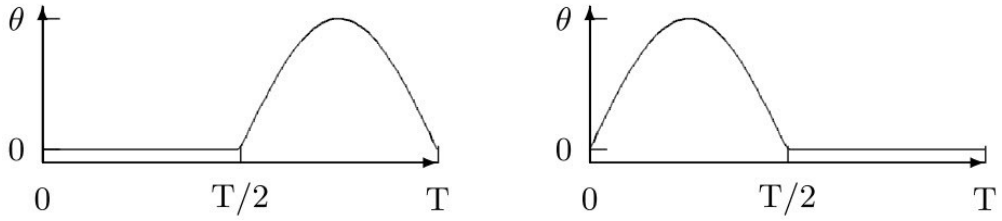

Figure 5.2: Left leg and right leg hips angular trajectories

Considering the illustrated walking sequence, Figure 5.2 can be assumed as the hip angular trajectory in one period of walking. It is a sinusoidal signal that has a lock phase at Zero degree. Therefore, in order to produce the proper angular trajectories to move the hips in coronal plane, proper parameters for the equation (5.3) should be obtained.

$$\theta_{hy} = A_{hy}\, sin(\omega t), if\ (t < T_{hy}/2)$$
$$\theta_{hy} = 0\,, otherwise \tag{5.3}$$

Where, A_{hy} and ω are assumed as the amplitude and the frequency of the signal, respectively, and T_{hy} is assumed a period of hip angular trajectory in coronal plane. As mentioned before, ankle trajectories can be calculated from hip trajectories. As shown in Figure 5.2, left and right hip angular trajectories are the same but with a phase shift of π. The period of walking in sagittal plane and coronal plane is equal. Therefore T_h and ω are eliminated and to produce the proper abduction and adduction, the proper value of A_{hy} parameter should be found.

5.2.2 Turn-in-Place Modeling

In this section, a new model-free approach to generate turn-in-place motion is proposed, in which TFS are used to model joints angular trajectories. A biped robot needs to move legs joints in three planes, namely transverse, coronal and sagittal, in order to perform turn-in-place.

Kajita et al. presented an approach to generate a turning gait namely forward-turning, in which the objective of turning is to follow a curve with certain radius [153]. Forward-turning and turn-in-place are different skills. The aim of turn-in-place is to change the robot direction at a fixed position. Combining turn-in-place and forward walking enables the robot to avoid obstacles with different shapes. Although the turn-in-place has been used as a basic motion for humanoid robots, there are few previous works on it [154]. In 2007, Tang et al. presented a model-based approach for generating turn-in-place motion using the Linear Inverted Pendulum Model (LIPM) and preview control of the ZMP [154].

We aim to generate a turn-in-place by using a model-free approach. We started by analyzing angular trajectories results from the previous work based on the LIPM approach, which can be found in [154]. The Angular trajectories of hip and knee, in sagittal, frontal and transverse plane, from this work are shown in following figures.

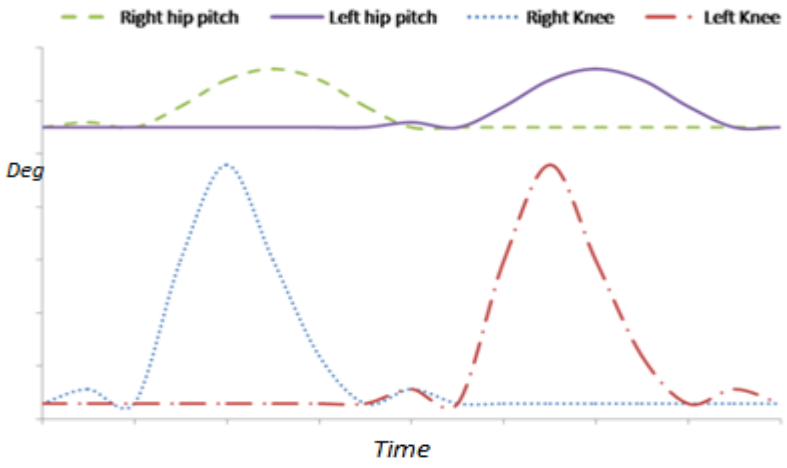

Figure 5.3: Hip and knee angular trajectories in sagittal plane [154]

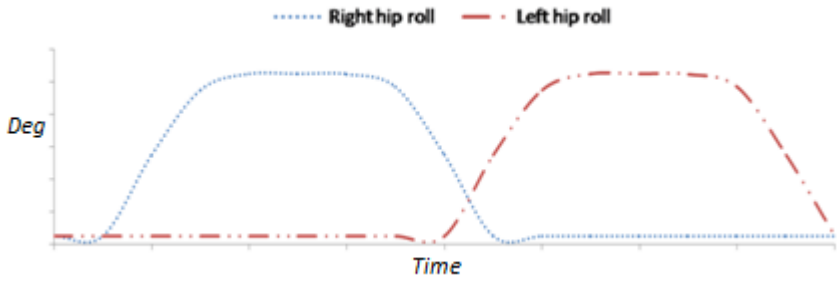

Figure 5.4: Hip angular trajectories in coronal plane [154]

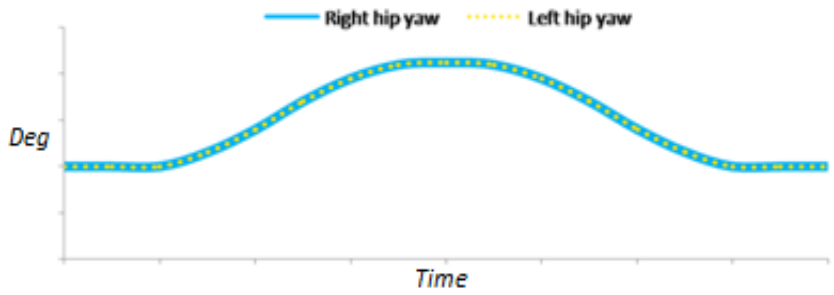

Figure 5.5: Hip angular trajectories in transverse plane [154]

The proposed approach for generating angular trajectories in sagittal plane is similar to the approach presented in section 5.2.1 for modeling the forward walk. In our turn-in-place modeling, similar to our forward walk modeling, foot in sagittal plane was kept parallel to the ground and ankle DoF parameters are eliminated. According to Figure 5.3, each angular trajectory has an offset, and the trajectories for both legs are identical in shape but are shifted in time relative to each other by half of the turning period. The TFS for generating hip and knee angular trajectories in sagittal plane are formulated in equation (5.4).

$$\theta_{hx} = A_{hx}\, sin(\omega_{hr} t) + C_{hx}\,, \omega_{hx} = \frac{2\pi}{T_h}$$
$$\theta_{hx} = C_{hx}, if(\theta_{hx} < C_{hx}) \tag{5.4}$$
$$\theta_k = A_k\, sin(\omega_k t) + C_k\,, \omega_k = \omega_{hx}$$
$$\theta_{hx} = C_{hx}, if(\theta_k < C_k)$$

In equation (5.4), A_{hx} and A_k are constant coefficients for generating signals. The h_x and k index stands for hip and knee in sagittal plane respectively. Also C_{hx} and C_k are signal offsets and T_h is assumed the period of hip trajectory. As mentioned in section 5.2.1, all movements except coronal movement have equal movement frequency and the frequency of coronal movement is half of other movements. Therefore, the $W_{hx=}\ W_{k=}\ 2\pi/T_{hx}$ equation can be derived. Fourier series parameters in sagittal plane are $A_{hx,}\ A_k$, $C_{hx,}\ C_k$ and $W_{hx.}$.

Each leg has two DOFs that move in coronal plane, which are hip yaw and ankle yaw. The hip yaw angular trajectory in a single turning period was shown in Figure 5.4. Similar to sagittal plane, foot stays parallel to the ground. To do so, angle of ankle yaw stays equal to hip yaw angle of the opposite leg. So the trajectory of ankle in frontal plane can be derived from the trajectory of hip angle in frontal plane.

Because of the similarity between the trajectories shapes in Figure 5.4 and Figure 5.2, we use the modeling approach for trajectory generation of coronal movement for Figure 5.4 the same as for Figure 5.2. Thus, The TFS model for generating hip trajectories in coronal plane for turn-

in-place is the same as presented in equation (5.3), in which the proper value of A_{hy} parameter should be found in a learning scenario.

The hip roll joint allows moving the leg on the transverse plane. As shown in Figure 5.5, for both legs the hip signal for this joint is the same, so we only need to determine the parameters of one leg. The TFS equations that generate hip trajectory in the transverse plane is formulated in equation (5.5).

$$\theta_{hr} = A_{hr}\, sin(\omega_{hr} t) + C_{hr}\,, \omega_{hr} = \frac{\omega_{hx}}{2} = \frac{\pi}{T_h}$$
$$\theta_{hr} = C_{hr}, if(\theta_{hr} < C_{hr}) \tag{5.5}$$

In the equation (5.5), A_{hr} is constant coefficient for generating signal. The *hr* index stands for the hip role joint. C_{hr} is signal offset. As it was mentioned, the movement frequency of TFS for hip yaw joint is half of other joints, so the $W_{hr}= W_{hx}/2$ equation can be derived. Therefore, W_{hr} can be achieved from $W_{hx,}$ and it is eliminated from parameters to be found. Consequently, A_{hr} and C_{hr} should be determined.

The parameters of our presented model to generate angular trajectories in all planes for generating turn-in-place motion are $A_{hx,}$ $A_{hy,}$ $A_{hr,}$ A_k, C_{hx} ,$C_{hr,,}$ C_k and W_{hx}. In this case, an optimization algorithm must optimize the 8 dimensions problem to find the best turn gate generator.

5.3 Reducing the Role of Inertia

Human walking is a complex activity which consists of four main sub-tasks [155]:

1) The initiation and termination of locomotors movements

2) The generation of continuous movement to progress toward destination

3) Maintenance of equilibrium during progression

4) Adaptability to meet any changes in the environment

In this section we will describe the first item (the initiation and termination of locomotors movement), but before introducing the solution, an overview on inertia is needed. “Inertia is the resistance of an object to a change its state of motion”. In the initiation and termination of locomotors movements, the robot changes its state from stopping to moving and vice versa, thus the inertia effect can be seen in these situations.

In order to reduce the role of inertia, a walking mechanism must increase speed smoothly. It was shown that humans try to increase his step length when they want to reach a higher speed

[156]. It was also shown that when the amplitude of angular trajectory is increased, the gaits (walking steps) get bigger and walking becomes faster [105]. Therefore, by increasing the walking speed and amplitude linearly, from standing to running, the effect of inertia will be reduced and the robot can walk more stable in the beginning of its walk. A model for the robot to walk from smaller gait with lower speed to bigger gait with higher speed was presented in 2009 [105], in which a linear equation is used to lead the robot to increase the amplitude of trajectory linearly from zero (stop state) to amplitude of a desired angular trajectory. Arms and legs angular trajectories in sagittal plane, which was presented in section 5.2.1, will be multiplied by the product of the equation (5.6).

$$\begin{aligned} k &= \text{time}/T, \text{time} < T \\ k &= 1, \text{time} >= T \end{aligned} \tag{5.6}$$

Here T is assumed as a parameter to determine how much time is needed for this increment algorithm to reach these desired trajectories. According to the fact that movement in coronal plane decrease at higher speeds of walking, the movement of legs in coronal plane must be reduced when the speed of walking is increased [16]. Therefore, angular trajectories in coronal plane that is generated by the equation (5.6) will be multiplied by the reverse value of calculated k.

5.4 Gait Learning

The model proposed in subsection 5.2.1 for generating forward walking has some parameters which values should be found. In subsection 5.4.1, a learning scenario will be presented in which Particle Swarm Optimization (PSO) is applied to find the best parameters values for generating a fast forward walk.

The TFS model proposed in subsection 5.4.2 for generating turn-in-place has also unknown parameters values. In order to learn the value of these parameters a turn learning scenario presented in subsection 5.4.2 in which Genetic Algorithm (GA) is applied to generate turn-in-place. All learning scenarios presented in this section are applied in a simulation environment.

5.4.1Forward Walk Learning

According to section 5.2.1, for producing forward walk movements in sagittal plane, TFS has six parameters to generate legs joints angular trajectories and one parameter to swing arms. The model to increase the walk speed in the beginning of the walk also has one parameter which was presented in section 5.3. There is also one parameter to produce proper legs movement in coronal plane. Therefore, in forward walk learning scenario, there is a 9-dimension search space

for the PSO to find the optimum solution. As mentioned in section 4.2.2, the parameters of the problems are coded into a finite length string as a particle in PSO algorithm.

To learn how to walk fast, angular trajectories produced by each particle are applied to a simulated robot. To use angular trajectory for walking, all individual robot's joints attempt to drive towards their target angles using Proportional Derivative (PD) controllers. To enable the robot with a fast walking skill, a fitness function based on robot's straight movement in limited action time is considered. First the robot is initialized in x=y=0 and it walks for 15 seconds. After that, the fitness function is calculated whenever the robot falls or the time duration for walking is over. The fitness function formulation is simply expressed as the distance travelled by the robot along the *x* axis, and if the robot fell the punishment equal to two meters would be subtracted from the calculated fitness.

Due to the fact that there is noise in simulated robot's actuators and sensors, training walking task in this approach can be viewed as an optimization problem in a noisy and nondeterministic environment. Resampling is one of the techniques to improve the performance of optimization algorithms in noisy environments [157]. In Resampling, for each solution, the fitness $F(y_i)$ is measured *m* times and yielding an averaged fitness $\overline{F(y_i)}$. According to equation (5.7) the noise strength of $\overline{\sigma_e}$ is reduced by a factor $\sqrt{m}$. In this study, *m* is assumed as 3.

$$\overline{F(y_i)} = \frac{1}{m}\sum_{k=1}^{m} F(y_i)\,, y_i = const. \Longrightarrow \overline{\sigma_e} = \sqrt{Var[\overline{F(y_i)}]} = \frac{\sigma_e}{\sqrt{m}} \tag{5.7}$$

Since particles may not satisfy some constraints after updating position procedure, constraint handling is a vital step in PSO algorithm. There is an obvious constraint on parameters in this learning scenario, i.e. time parameters in TFS should be positive. Therefore, Pareto [24] with multi-objective modeling is used for handling constraints.

In Pareto, a solution, *x(2)*, is dominated by solution, *x(1)*, if *x(1)* is not worse than *x(2)* in all objectives, and for at least one of the objectives, *x(1)* is strictly better than *x(2)*. Without loss of generality, these conditions can be expressed as follows for the case where all of the objective functions are to be minimized:

$$fm\big(x(1)\big) \leq fm\big(x(2)\big)\ for\ \forall m = 1,2,\dots,M \text{ and}$$

$$fm\big(x(1)\big) < fm\big(x(2)\big)\ for\ some\ m.$$

Each constraint is assumed as an object in which parameters must be satisfied. According to Pareto method, a particle can be considered to calculate the value of P_i, P_k^g in equation (4.3), when it satisfies objects and constraints. Therefore, calculating fitness for particles that cannot satisfy constraint is not necessary.

We have considered various values for each parameter of the PSO presented in equation (4.3) and tried all possible combinations. Finally we chose the values of the parameters such that C_1 and C_2 and Δt are assumed as 1, 1.5, and 1, respectively. The parameters of the dynamic weights method given in equation (4.6) including w_{init} and U are assumed as 0.8 and 1.0002, respectively. We have also used for our experiments a particle swarm composed by 100 particles (N = 100) and a maximum iteration of 10.

5.4.2 Turn-in-place Learning

Robot turning is a complex motion because many factors affect turning style and stability such as robot kinematics, collision between feet and the ground and dynamics of the robot. For this complex motion, relation between turning gait trajectory and turning characteristic is nonlinear. Therefore, optimizing and learning a turning gait is usually difficult.

In this section, GA that was explained in section 4.2.1 is used to find the best parameters to generate angular trajectories for robot turning motion. Parameters are coded into a finite length of string (Genes) as a chromosome. According to section 5.2.2, Truncated Fourier Series (TFS) has 8 parameters to generate all joints angular trajectories, thus each chromosome has 8 Genes. In designing the chromosomes, gene type is considered as double format. Population for each generation is assumed to be 100. Chromosomes are generated randomly and uniformly for the first iteration between lower and upper bound. In this learning scenario, the lower and upper bound values are given in the Table 5.1.

Table 5.1: Lower bound and upper bound

	Chx	*Ck*	*Chr*	*Ahr,*	*Ak*	*Ahy*	*Ahx,*	*Whx*
Upper Bound	30	0	0	60	0	40	45	1
Lower Bound	-10	-50	-45	20	-60	0	0	0.05

Fitness function has a critical role in GA and is used to judge whether a solution represented by a chromosome is good or bad. To learn how to turn, first angular trajectories based on each chromosome are produced by TFS, and then these angular trajectories are used by simulated robot for turning. To execute these angular trajectories all individual robot joints are controlled by using PD controllers.

To achieve more stable and faster turn-in-place, a fitness function based on robot's turn-in-place motion with limited time is assumed. The amount of deviation from the initial position *(dist)* is subtracted from the maximum of the degree which robot could turn, to force the robot to turn stably and in place. The fitness function is calculated when the robot falls or when time limit is reached. Equation (5.8) shows the fitness function formulation, where the robot is initialized in

x=y=0 (0,0) aligned to the horizontal axis where *rotdegree=0* and time duration for turning is 60 seconds.

$$Fitness = rotdegree - 90*dist \quad (5.8)$$

In equation (5.8), the *rotdegree* is the amount of turning at radian and *dist* is the amount of linear movement from initial place. In this learning scenario, the GA optimization has been configured with a scattered function for the cross over operator. For mutation, a uniform function has been used with mutation rate is assumed equal to 0.06. Selection method is roulette wheel and reproduction rate assumed equal to 0.8. Termination condition is having a generation counter greater than 28. Therefore, by considering the population size (which is 100), the GA requires 2800 trials to find appropriate TFS parameters.

5.5 Adapting Learned Trajectories to Apply on Real Robots

The gait optimization and learning scenarios presented in the previous section were applied in simulation environment, which let the simulated robot learn how to walk forward or turn-in-place. Every physical simulator contains some simplifications in its real world model; therefore, the results of simulation and reality are usually not the same. This issue is called the reality gap. In this section, an approach is presented to remodel the learned trajectories achieved in simulation to apply them on the real robot.

The reality gap is smaller if the biped locomotion is executed at slow speeds; also the stability of them is enhanced when using lower speeds. According to [105] smaller gait with lower amplitude has lower speed and acceleration than bigger gait with higher amplitude. In order to reduce the speed of turn-in-place or forward walk, all angular trajectories of the legs that has been learned in simulation can be multiplied by a variable assumed as *K*. The value of this variable can range from zero to one. When *K* is equal to zero, the lowest speed will be produced and the robot will be stopped. When *K* is equal to one, the same trajectories obtained in simulation will be produced. The proper value of *K* can be found by trial and error method.

In our experiment, reducing the speed of the locomotion skills is not sufficient to be used directly on a real robot. The foot trajectory should also be remodeled to reduce the role of friction. The specification of TFS approach, by defining each joint angular trajectory independently from the others, does not provide an adequate model for controlling the foot trajectory. To achieve a more controllable model of foot trajectory, the TFS approach specification is converted to use the leg interface.

In 2006, the leg interface was presented in a biped walking approach, in which it was used to control the leg movements [158]. The leg interface allows specification of the leg positions by using three components: leg angle, leg extension and foot angle.

The leg angle θ_{leg} is the angle between the torso and the line connecting hip and ankle, and is given as: $\theta_{Leg} = (\theta^r_{Leg}, \theta^p_{Leg}, \theta^y_{Leg})$ where θ^r_{Leg} , θ^p_{Leg} and θ^y_{Leg} are roll, pitch, and yaw angles respectively. When the leg is parallel to the trunk, θ^r_{Leg} is equal to zero, and when the leg is moved outward to the side in coronal plane, θ^r_{Leg} is greater than zero. In sagittal plane, when the leg is parallel to the trunk $\theta^p_{Leg} = 0$, and when the legs moves to the front $\theta^r_{Leg} > 0$.

Leg extension γ denotes the distance between the hip and the ankle. It can be normalized in the range of -1 and 0 , $-1 \leq \gamma \leq 0$, where $\gamma = -1$ denotes that the leg is fully extended and $\gamma = 0$ denotes that the leg is fully shortened.

Using the leg interface, the foot height trajectory can be remodeled by just changing the value of leg extension parameter. Changing the leg extension value of the support foot, the robot can produce the movement of the CoM in *Z* axis. This type of movement can produce a little jumping and increase in foot height trajectory, which diminishes foot collisions with the ground and results in more walking or turning behavior. The Fourier Series (FS) formulation to produce leg extension trajectory is given in equation (5.9).

$$\gamma = A_{ex} . sin(w_{ex} t) + C_{ex} \quad W_{ex} = W_{hx} \tag{5.9}$$

Where, the *ex* index stands for extension, A_{ex} is constant coefficient for generating signal and C_{ex} is signal offset. Like in the previous equations, the movement frequency of FS is equal to other behaviors, so the $W_{ex} = W_{hx}$. A_{ex} and C_{ex} are the two parameters of the model of the leg extension generator, which the value must be found. The value of these two parameters are also calculated by trial and error method.

5.6 Results

The TFS based approaches, which were presented in this section, were tested on a NAO humanoid robot. The specification of the NAO robot used in this study is given in section 1.4. In this section, first, the results achieved by learning scenarios in simulation for generating forward walk and turn-in-place are presented, in which experiments were performed using Simspark simulator. The Simspark was addressed in section 4.3.1. Then the result achieved by a real robot is given.

5.6.1 Forward Walk Learning Results

In this section, the results of the presented leaning scenario for forward walk generation are explained. To compare the proposed method that can produce legs movement in coronal plane with basic TFS method that can produce the joints movement only in sagittal plane, both of these approaches were tested using the same simulator and the same machine specification. We optimized both methods by utilizing PSO with specifications that was presented in section 5.6.1.

Basic TFS model which generates walking movements based on the model presented in section 5.2.1 without considering coronal movement has 8 parameters, including $A_{hx,}$ A_k , $C_{hx,,}$ C_k and W_{hx} in equation (5.1), A_{arm} in equation (5.2), A_{hy} in equation (5.3) and T in equation (5.6). The values of these parameters are optimized by PSO algorithm running on a Pentium IV 3 GHz Core 2 Duo machine with 2 GB of physical memory. After, performing 3000 trials in 4 hours, the robot could walk 8.6 m in 15 s with average body speed of 0.57 m/s and period of 0.41 s for each step. Figure 5.6 shows the average and best fitness values during these 10 generations.

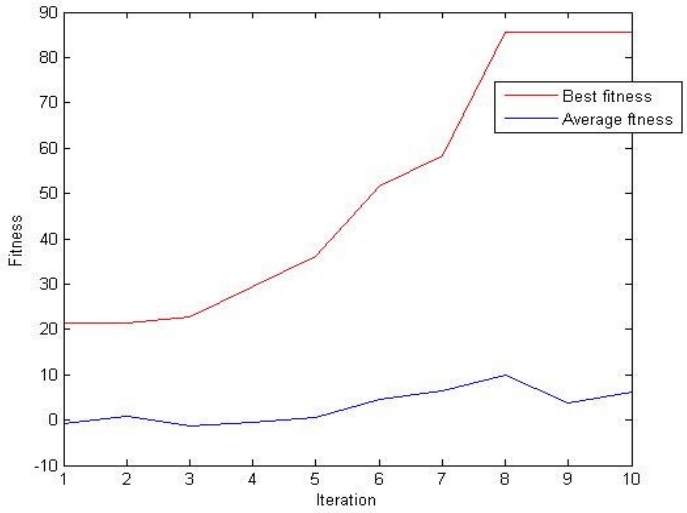

Figure 5.6: PSO convergence for previous TFS

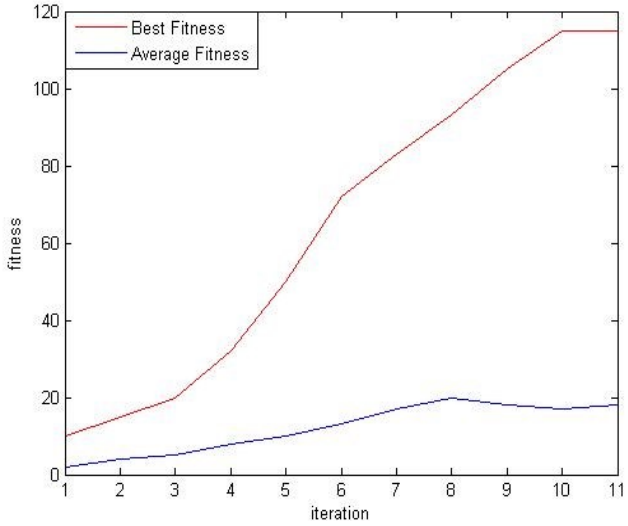

Figure 5.7: PSO convergence for new approach and TFS model

Using the proposed approach to model the forward walk with coronal movement, after 3300 trails and 5 hours from starting PSO in a machine with the same specifications, the robot could walk 11.5 m in 15 s. Average and best fitness are shown in Figure 5.7.

Robot could walk with average body speed of 0.766 m/s by using the TFS with coronal movement. For this walk, the learned sagittal trajectories of left hip and knee after robot started to walk are shown in Figure 5.8. From this figure, it can be noticed that the robot increased its speed in 1.7 seconds.

The learned trajectory of left arm is also shown in Figure 5.9. Finally, the learned trajectory of left hip in coronal plane for abduction and adduction movements is also shown in Figure 5.10.

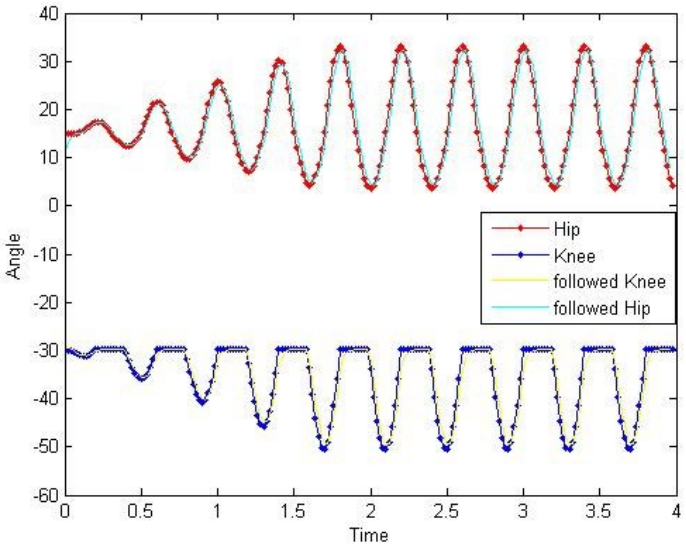

Figure 5.8: Left hip and knee trajectories in the sagittal plane

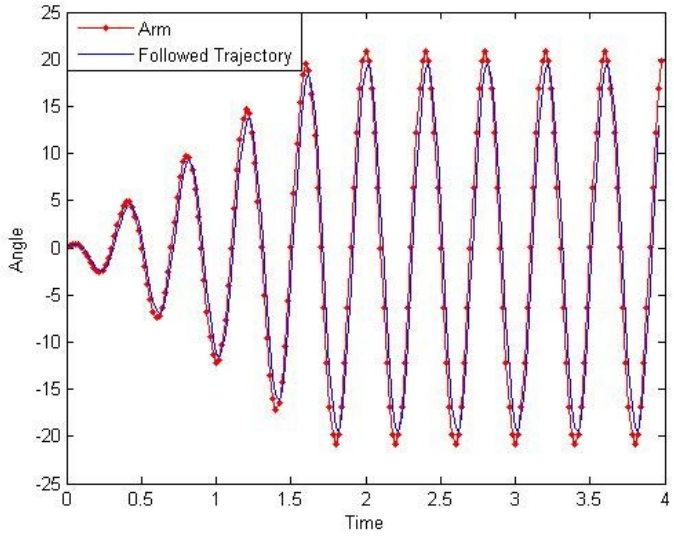

Figure 5.9: Left arm trajectory

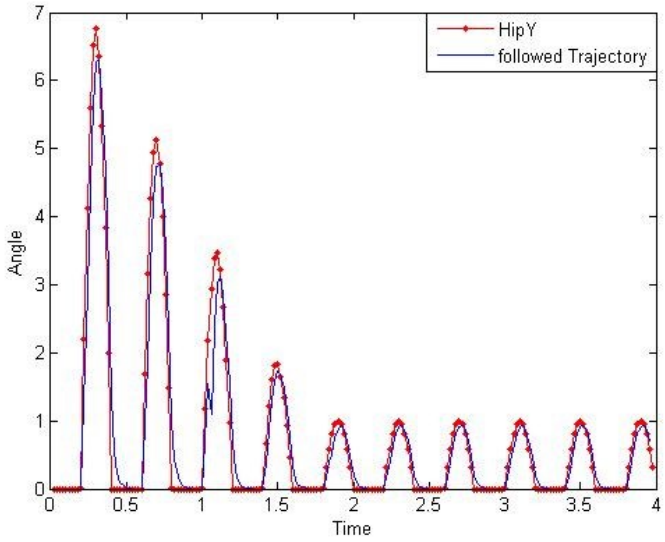

Figure 5.10: Left hip trajectory in coronal plane (hipY)

After the learning procedure, the robot could walk with the average speed of 0.77 m/s. The best results of walking behavior for the teams that participated in RoboCup 2010 were chosen for the comparison [25].

Table 5.2, presents the comparison of the best results of RoboCup teams, compared with the proposed approach. Analyzing the table it is clear that forward walking achieved by our approach, may outperform the same skill of all teams analyzed except SEU.

Table 5.2: Comparison the average speed for forward walking in different teams (m/s)

	Proposed approach	*FCPortugal*	*SEU*	*Wright Eagle*	*Bats*
Forward Walking	0.76	0.51	1.20	0.67	0.43

5.6.2 Turn-in-place Results

In this section, the result of the proposed learning scenario in section 5.6.2 for generating trun-in-place is presented. Five hours after starting GA on a Pentium IV 3 GHz machine with 2 GB of physical memory, 2800 trails was performed with 28 number of generations.

The learning procedure converged, such that the robot could turn about 360 degree in 4s with average body speed of around 90 degree/s. Figure 5.11 shows the average and maximum fitness values for the robot over 28 generations. Since physical simulation is not accurate and has noise. Maximum fitness was falling as well as rising from generation to generation.

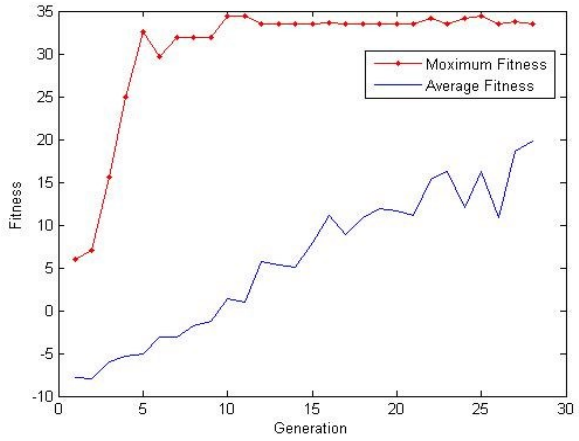

Figure 5.11: Average fitness and Maximum fitness during 28 generations

The angular trajectories that generated by learned TFS and followed by controller are shown in Figure 5.12.

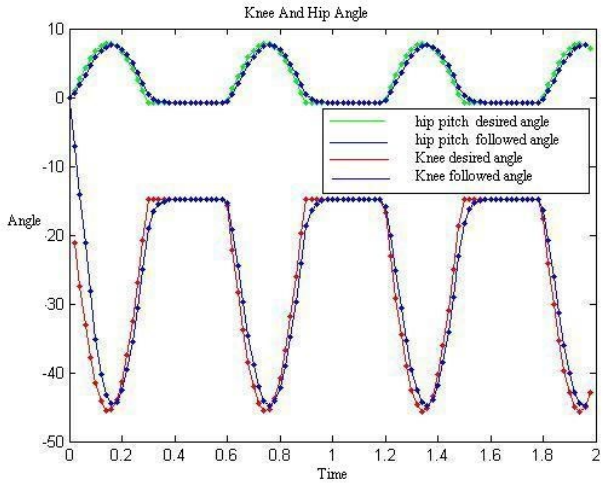

Figure 5.12: Angular trajectories generated by learned Fourier Series and followed by controller, for left hip and knee in the sagittal plane

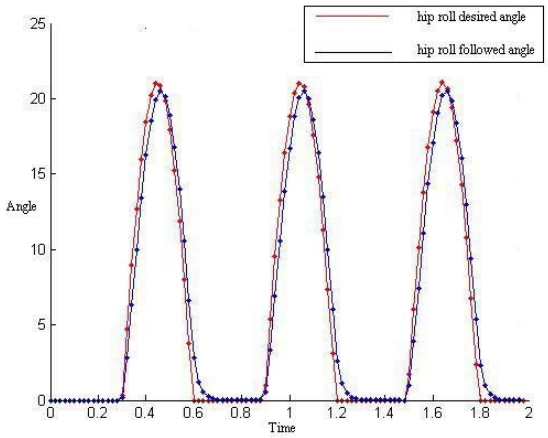

Figure 5.13: Left roll hip trajectory of the simulated NAO robot

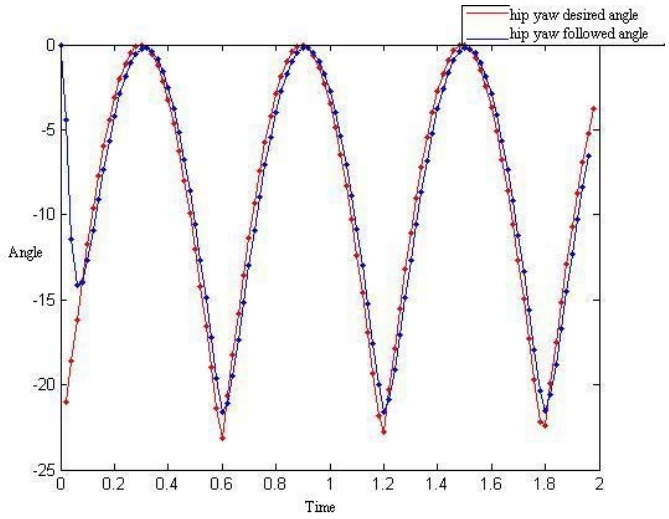

Figure 5.14: Left yaw hip angular trajectory

5.6.3 Real Robot Results

The model for remoding learned trajectories to be applied in real robots which was presented in section 5.5 has 3 parameters, including K for reducing the turn speed, A_{ex} and C_{ex} in equation (5.9). The proper values of these three parameters are found by trial and error. Turn-in-place on the real NAO was achieved with K value of 0.3. NAO robot could turn in place 360 degree in 8 seconds stably and average body speed was around 45 degree/s.

Figure 5.15-b shows turn-in-place obtained from GA search in simulation. Figure 5.15-a shows NAO robot while performing turn-in-place behavior after adaptation on the robot. Figure 5.16 also shows hip and knee angular trajectories generated by learned Fourier Series and followed by servo motors of the robots joint.

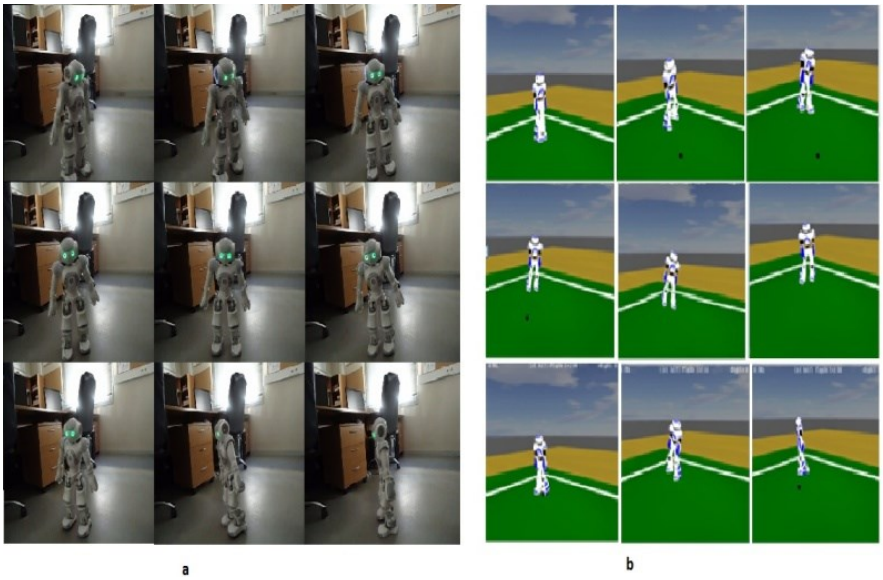

Figure 5.15: A) NAO robot was adapted to do turning by using simulation results in lower amplitude B) Simulated NAO robot follows the learned nominal trajectory to do turn-in-place

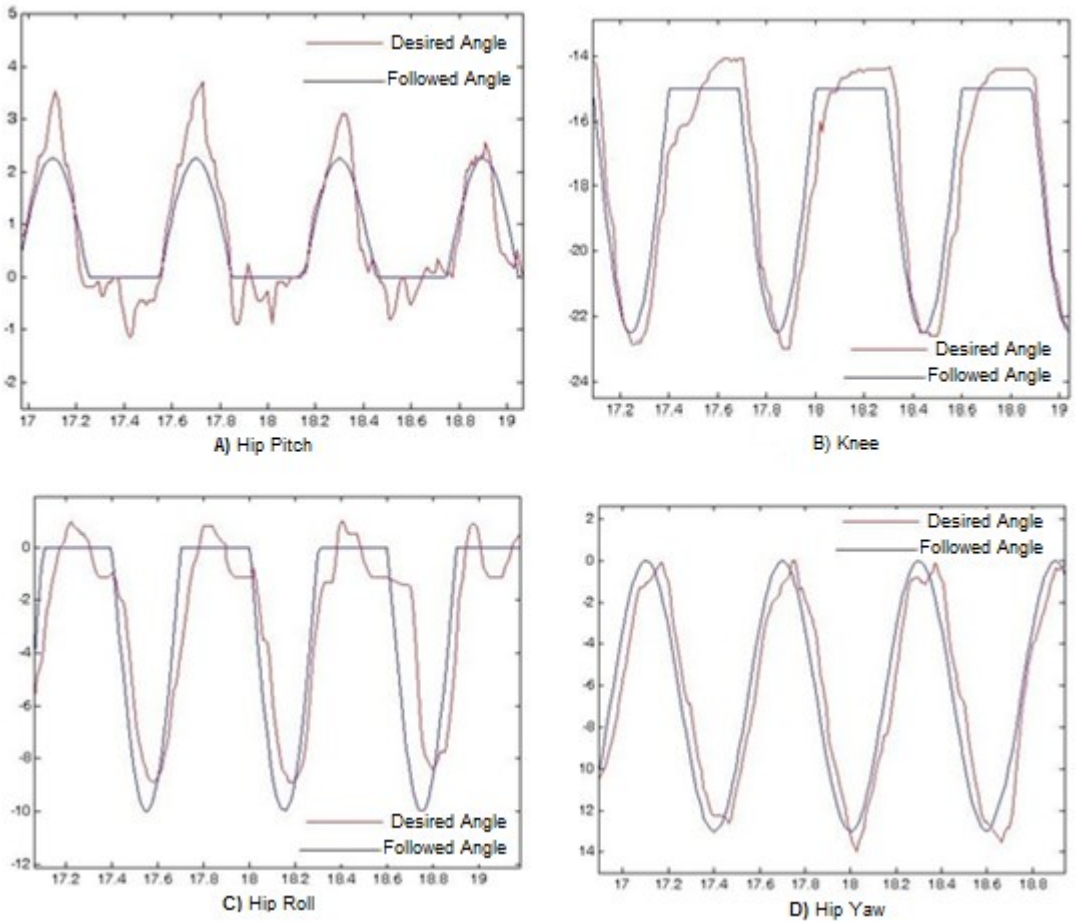

Figure 5.16: Angular trajectories generated by learned and adapted gait generator and followed by Nao Robots servo motors and PID controller

5.7 Summary

In this chapter, we present an approach to model forward walk and turn-in-place. In forward walk modeling section, The TFS model for forward walk is presented in which all joints movements in coronal and sagittal plane are used. An optimized TFS is used to produce leg and arm movements in sagittal plane. A new model was presented to generate legs walking movement in coronal plane. Our experimental results given in section 5.6.1 show that the using of joints movements in sagittal and coronal planes to compose the forward walking skill allowed the biped robot to walk faster than previous methods that only used the joints in sagittal plane.

In turn-in-place section, a novel model-free approach to generate turn-in-place motion has been presented. The proposed TFS based gait generator has 8 parameters for producing all joints angular trajectories. An optimized gait to produce the fastest possible turning was found using genetics algorithm. Finally, the NAO robot could perform turn-in-place using the results achieved by simulation.

The TFS based approach presented in this chapter have three main advantages. First, it can be used for humanoid robots without considering its physical model. Second, these approaches have a low computational complexity, which is suitable to be implemented on humanoid robots with low computational capabilities where the response of controller should be real time. Finally, Fourier series can parameterize trajectories well; therefore, robot locomotion controller designer can manipulate trajectories simply by changing parameters of Fourier series.

Despite the above advantages, the TFS based approaches have a general drawback, which is that they are not able to generate omnidirectional walking. In an omnidirectional walking, the robot should be able to walk in any direction and to change the walk direction in real-time. However, the TFS model should be remodeled, and its parameters should be learned for each walk direction. For example, as it was shown in section 5.2.1 and 5.2.2, two different angular trajectory modeling approaches are used to model the turn-in-place and forward walking. Since an omnidirectional walk includes many different walk directions, it is impossible to model all of walking directions by TFS.

Chapter 6

Cart Table Model Applied on Humanoid Robots

This chapter presents the results achieved by our attempt to implement walking approaches based on Cart-Table model, which led us to publish the following articles:

> **N. Shafii**, A. Abdolmaleki, N. Lau, L. P. Reis *" Development of an Omnidirectional Walk Engine for Soccer Humanoid Robots"* International Journal of Advanced Robotic Systems. (IF: 0.579) (Accepted)
>
> **N. Shafii**, A. Abdolmaleki, R. Ferreira, N. Lau, L.P. Reis *"Omnidirectional Walking and Active Balance for Soccer Humanoid Robot"*, Progress in Artificial Intelligence, Lecture Notes in Computer Science, vol. 8154, pp. 283-294, Springer, 2013.
>
> R. Ferreira, **N. Shafii**, N. Lau, L.P. Reis, A. Abdolmaleki *"Diagonal Walk Reference Generator based on Fourier Approximation of ZMP Trajectory"*, In Proceedings of the 13th International Conference on Autonomous Robot Systems (ICARSC), pp. 1-6, IEEE, 2013

6.1 Introduction

In the previous chapter, we proposed a model-free approach, in which a Truncated Fourier Series (TFS) was used. We concluded that it is not possible to generate an omnidirectional walk by using the presented TFS approach.

In this chapter, we will focus on model-based approaches which can generate omnidirectional walking. As it was mentioned in Chapter 3, the ZMP based approach is the most popular model-based methodology, which can create an omnidirectional walk. In this chapter, we will

present ZMP based approaches, in which the Cart-Table model (presented in section 3.2.1) is used to model the ZMP dynamics.

As it was presented in section 2.2, the ZMP dynamics gives the relation between the ZMP and CoM trajectory. This relation between ZMP and CoM can be simplified and approximated by using dynamics equations of motion of a simple physical system. Kajita et al. assumed that biped walking is a problem of balancing a Cart-Table model [61]. As it was mentioned in section 3.2.1, the main issue of applying Cart-Table model is how to solve its differential equations. Even though CoM trajectory can be calculated theoretically by using the exact solution of the Cart-Table differential equations, applying the calculated trajectory is not straightforward in a real biped robot walking because the solution consists of unbounded functions *cosh*, and the obtained CoM trajectory is very sensitive to the time step variation of the walking gait.

The approaches presented previously, on how to tackle this issue, are organized into two major groups, optimal control approaches and analytical approaches. Kajita et al. have presented an approach to find the proper CoM trajectory, based on the preview control of the ZMP reference, which enables the robot to perform an omnidirectional walk [61]. The ZMP preview control approach is a dominant approach based on optimal control theory. In section 6.3.2, the ZMP preview control approach will be presented in detail.

The main goal of this chapter is to solve the Cart-Table differential equations analytically by using Fourier approximation of the ZMP. We will compare the proposed analytical approach with the ZMP preview control to show the benefit of using analytical approach. In the literature, there are two different approaches to apply Fourier approximation to solve the ZMP differential equation.

The first approach can only generate walking in a single direction, i.e., the robot can walk in a specific direction and cannot change its walking direction during its walk. A Fourier series approximation based method, which generates the CoM trajectory, was previously proposed for straight and curved walking [159] [160]. In section 6.2, we will explain how to improve this approach to generate diagonal walking. The diagonal walking reference trajectory generation method based on Fourier approximation will described in section 6.2.1 in detail.

The second type of the Fourier based approach can generate omnidirectional walking, this approach will be presented in section 6.3, in which time segmentation with Fourier approximation of the ZMP is used to solve the cart table model analytically.

The results achieved by using the two proposed walk engines will be presented in sections 6.4.1 and 6.4.2. Finally, in section 6.5, we will conclude the results of these two presented walk engines, and address the cons and pros of using cart-table model.

6.2 Single Direction Walk Engine

Cart-table model mainly, which was discussed in section 3.2.1, will be used in this chapter in order to generate walking. For applying cart-table model in a biped walking problem, first the position of the foot during walking must be planned and defined. Then, based on the position of the support polygon, the ZMP trajectory can be designed. In the next step, the position of the CoM from differential equations (3.2) (3.3) must be calculated. Finally, inverse kinematics is used to find the angular trajectories of each joint based on the planned position of the foot and calculated CoM. The main issue of applying Cart-table model is how to solve the aforementioned differential equations.

Erbatur et al. presented a method for generating the trunk motion by transforming ZMP reference trajectories into the Fourier series. This approach could create straight walking reference [70][159]. Recently this reference generation technique has been improved, and tested on the real humanoid robot to generate curved walking [160].

Although to enable a humanoid robot for walking in a straight or curved line is very important and motivating for researchers, generating other types of walking such as diagonal walking can improve the ability of a humanoid robot to avoid obstacles. To the best of our knowledge, there is no method based on this type of Fourier approximation approach to generate diagonal walking. Therefore, the contribution of this section is to extend the Fourier based straight walking approach in [70] [71] into generating a diagonal walking. In order to create diagonal walking with desired speed, a combination of straight and side walking is required. The speeds on X and Y directions can be varied according to user input. In order to test its performance, the proposed walking reference generation system is applied to the NAO simulated robot.

6.2.1 Fourier Series Approach for Generating Diagonal Walk

In this section the work in [70] is extended to endow the robot with a diagonal walk. First, the reference trajectory for the ZMP is formulated, and then it is approximated by using Fourier series. Finally, the position of the CoM is obtained by solving the differential equations of the cart-table dynamics that was presented in (3.2) (3.3), in which the source term P is assumed as the approximated ZMP.

In order to determine the reference trajectory of the ZMP, first the ZMP trajectory in a normal diagonal walk is analyzed. Figure 6.1 shows the position of feet, on the ground plane, over t seconds.

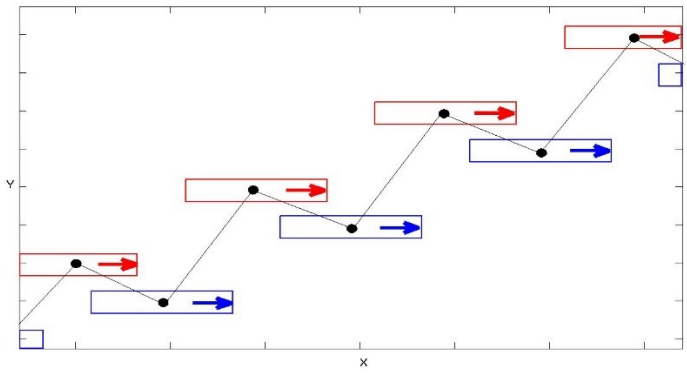

Figure 6.1: Footsteps of a diagonal walking

To better understand the ZMP trajectory, ZMP movement is decomposed along *X* and *Y* axis, giving the advantage to illustrate them as functions of time. Figure 6.2 and Figure 6.3 show the ZMP trajectories of X and Y, respectively.

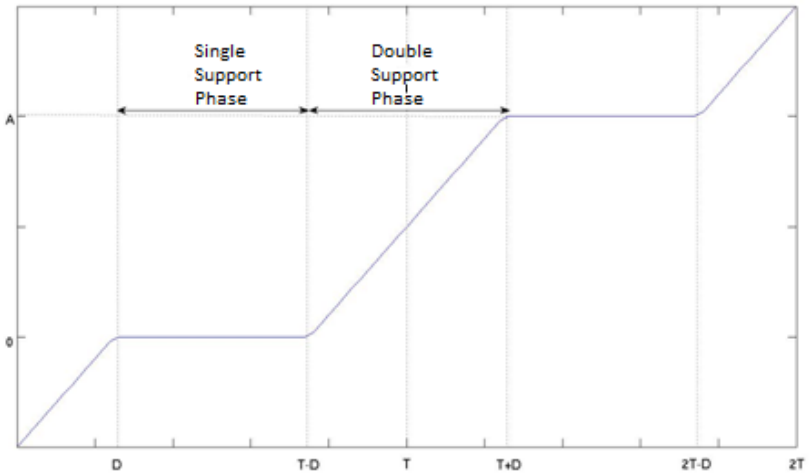

Figure 6.2: ZMP trajectory in X direction

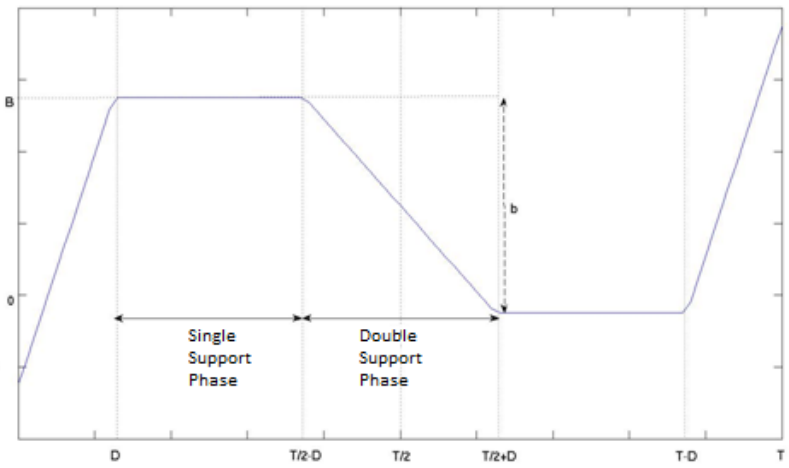

Figure 6.3: ZMP trajectory in Y direction

ZMP trajectories can be seen as the combination of periodic and non-periodic component. Figure 6.4 and Figure 6.5 illustrate the periodic and non-periodic components of the ZMP trajectories in the *X* and *Y* direction, respectively.

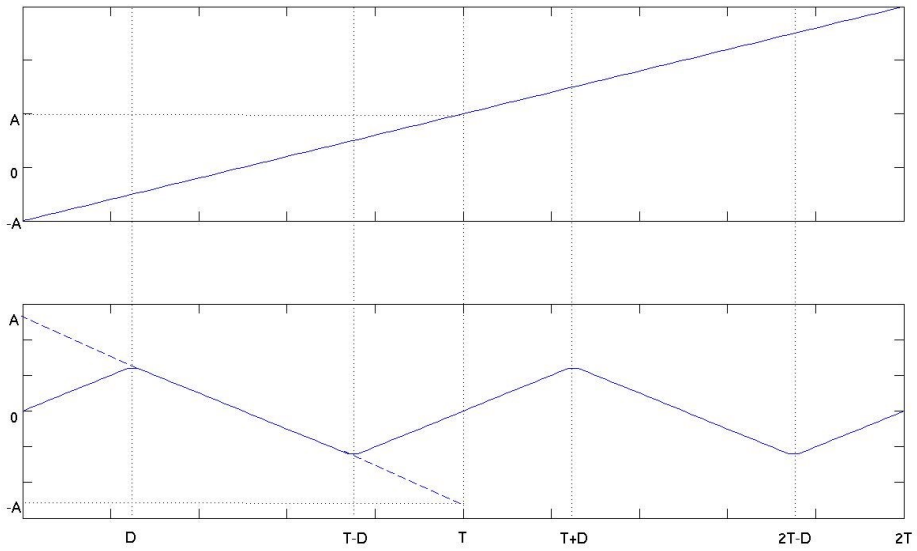

Figure 6.4: ZMP component in X direction

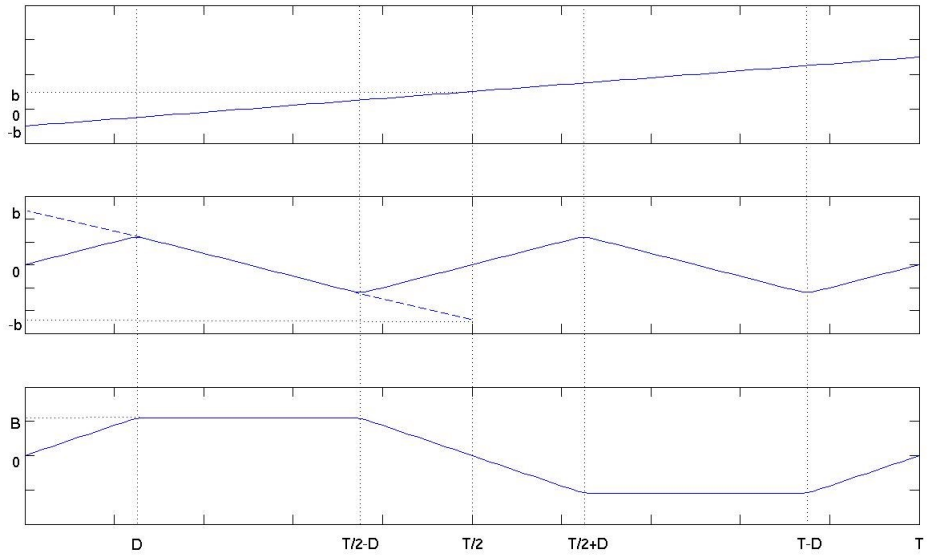

Figure 6.5: ZMP component in Y direction

In order to employ Fourier series limited to a certain number of terms, continuous periodic functions are needed. Figure 6.4 and Figure 6.5 are employed to determine the equations (6.1) and (6.6) for formulating the ZMP reference trajectory in X and Y direction respectively. The equations from (6.3) to (6.5) and from (6.8) to (6.17) formulate periodic components of the

trajectory. The non-periodic component of the ZMP trajectories are presented in equations (6.2) (6.7).

This formulation design enhances the approach presented in [70] by adding a double support phase to formulate the ZMP trajectory, allowing to control the amount of double support phase using the parameter *D*. The parameters used in the ZMP trajectory formulation are listed in Table 6.1 as well as a brief description of each one.

Table 6.1: Formulation parameters

Parameter	*Description*
T	Period
A	Amount of displacement in X direction
B	Distance between both feet when walking straight
b	Amount of displacement in Y direction
D	Amount of time to be in double support phase

$$\mathrm{ZMP_x^{ref}(t)} = \mathrm{f_{x1}(t)} + \sum_{\mathrm{i=1}}^{3} \mathrm{f_{2\_i}(t)} \tag{6.1}$$

$$\mathrm{f_{x1}(t)} = \frac{\mathrm{A}}{\mathrm{T}}\left(\mathrm{t} - \frac{\mathrm{T}}{2}\right) \tag{6.2}$$

$$\mathrm{f_{x2\_1}(t)} = \mathrm{A}\left(\frac{1}{2\mathrm{D}} - \frac{1}{\mathrm{T}}\right).\mathrm{t}.\left(\mathrm{u(t)} - \mathrm{u(t-D)}\right) \tag{6.3}$$

$$\mathrm{f_{x2\_2}(t)} = \mathrm{A}\left(\frac{1}{2} - \frac{1}{\mathrm{T}}\right)\left(\mathrm{u(t-D)} - \mathrm{u(t-(T-D))}\right) \tag{6.4}$$

$$\mathrm{f_{x2\_3}(t)} = \mathrm{A}\left(\frac{1}{2\mathrm{D}} - \frac{1}{\mathrm{T}}\right).(\mathrm{t}-\mathrm{T}).\left(\mathrm{u(t-(T-D))} - \mathrm{u(t-T)}\right) \tag{6.5}$$

$$\mathrm{ZMP_y^{ref}(t)} = \mathrm{f_y(t)} + \sum_{\mathrm{i=1}}^{5} \left(\mathrm{f_{y2\_i}(t)} + \mathrm{f_{y3\_i}(t)}\right) \tag{6.6}$$

$$f_{y1}(t) = \frac{b}{T}\left(t - \frac{T}{2}\right) \tag{6.7}$$

$$f_{y2\_1}(t) = \frac{2b\left(\frac{T}{4} - D\right)}{DT}.\mathrm{t}.\left(\mathrm{u(t)} - \mathrm{u(t-D)}\right) \tag{6.8}$$

$$\mathrm{f_{y2\_2}(t)} = -\frac{2\mathrm{b}}{\mathrm{T}}.\left(\mathrm{t} - \frac{\mathrm{T}}{4}\right).\left(\mathrm{u(t-D)} - \mathrm{u}\left(\mathrm{t} - \left(\frac{\mathrm{T}}{2} - \mathrm{D}\right)\right)\right) \tag{6.9}$$

$$f_{y2\_3}(t) = \frac{2b\left(\frac{T}{4} - D\right)}{DT}.\left(\mathrm{t} - \frac{\mathrm{T}}{2}\right).\left(\mathrm{u}\left(\mathrm{t} - \left(\frac{\mathrm{T}}{2} - \mathrm{D}\right)\right) - \mathrm{u}\left(\mathrm{t} - \left(\frac{\mathrm{T}}{2} + \mathrm{D}\right)\right)\right) \tag{6.10}$$

$$f_{y2\_4}(t) = -\frac{2b}{T}.\left(\mathrm{t} - \frac{3\mathrm{T}}{4}\right).\left(\mathrm{u}\left(\mathrm{t} - \left(\frac{\mathrm{T}}{2} + \mathrm{D}\right)\right) - \mathrm{u}\big(\mathrm{t} - (\mathrm{T} - \mathrm{D})\big)\right) \tag{6.11}$$

$$f_{y2\_5}(t) = \frac{2b\left(\frac{T}{4} - D\right)}{DT}.(\mathrm{t} - \mathrm{T}).\Big(\mathrm{u}\big(\mathrm{t} - (\mathrm{T} - \mathrm{D})\big) - \mathrm{u}(\mathrm{t} - \mathrm{T})\Big) \tag{6.12}$$

$$f_{y3\_1}(t) = \frac{B}{D}.\mathrm{t}.\big(\mathrm{u}(\mathrm{t}) - \mathrm{u}(\mathrm{t} - \mathrm{D})\big) \tag{6.13}$$

$$f_{y3\_2}(t) = B\left(\mathrm{u}(\mathrm{t} - \mathrm{D}) - \mathrm{u}\left(\mathrm{t} - (\frac{\mathrm{T}}{2} - \mathrm{D})\right)\right) \tag{6.14}$$

$$f_{y3\_3}(t) = -\frac{B}{D}.\left(\mathrm{t} - \frac{\mathrm{T}}{2}\right).\left(\mathrm{u}\left(\mathrm{t} - \left(\frac{\mathrm{T}}{2} - \mathrm{D}\right)\right) - \mathrm{u}\left(\mathrm{t} - \left(\frac{\mathrm{T}}{2} + \mathrm{D}\right)\right)\right) \tag{6.15}$$

$$f_{y3\_4}(t) = -B\left(\mathrm{u}\left(\mathrm{t} - \left(\frac{\mathrm{T}}{2} + \mathrm{D}\right)\right) - \mathrm{u}(\mathrm{t} - (\mathrm{T} - \mathrm{D}))\right) \tag{6.16}$$

$$f_{y3\_5}(t) = \frac{B}{D}.(\mathrm{t} - \mathrm{T}).\Big(\mathrm{u}\big(\mathrm{t} - (\mathrm{T} - \mathrm{D})\big) - \mathrm{u}(\mathrm{t} - \mathrm{T})\Big) \tag{6.17}$$

The next step is to determine the Fourier series of equations (6.1) and (6.6). For this we used the definition of the Fourier series given by (6.18).

$$\mathrm{f(t)} = \frac{\mathrm{a_0}}{2} + \sum_{\mathrm{n=1}}^{\mathrm{N}} [\mathrm{a_n}\cos(\mathrm{nt}) + \mathrm{b_n}\sin(\mathrm{nt})]\ , \mathrm{N} \geq 1 \tag{6.18}$$

The results are given by equations (6.19) and (6.20) which are approximated ZMP trajectories using Fourier series definition (6.18).

$$ZMP_x^{fourier}(t) = \frac{A}{T}\left(t - \frac{T}{2}\right) + \sum_{n=1}^{N}\left[b_{n\_x}\,sin\left(\frac{2n\pi}{T}t\right)\right], N \geq 1 \tag{6.19}$$

$$ZMP_y^{fourier}(t) = \frac{b}{T}\left(t - \frac{T}{2}\right) + \sum_{n=1}^{N}\left[b_{n\_y}\,sin\left(\frac{2n\pi}{T}t\right)\right], N \geq 1 \tag{6.20}$$

Where

$$b_{n\_x} = \left(\frac{2A\sin\left(\frac{2n\pi}{T} \times DSP\right)}{T.DSP.\left(\frac{2n\pi}{T}\right)^2}\right) \tag{6.21}$$

$$b_{n\_y} = \left(\frac{2\,sin\left(\frac{2n\pi}{T} \times DSP\right)}{T.DSP.\left(\frac{2n\pi}{T}\right)^2}\right)\left((b - B)(-1)^2 + B + b\right) \tag{6.22}$$

Finally, it remains to solve the cart-table differential equations (3.2) and (3.3) using equations (6.19) and (6.20) as source term:

$$y - \frac{Z_h}{g}\ddot{y} = \sum_{n=1}^{N}\left[b_n \, sin\left(\frac{2n\pi}{T}t\right)\right], \quad N \geq 1 \tag{6.23}$$

The value of the parameter *N* must be a positive integer. The solution will be of the form:

$$y(t) = \sum_{n=1}^{N}\left[C_n \sin\left(\frac{2n\pi}{T}t\right)\right] \tag{6.24}$$

$$\ddot{y}(t) = \sum_{n=1}^{N}\left[-C_n\left(\frac{2n\pi}{T}\right)^2 sin\left(\frac{2n\pi}{T}t\right)\right] \tag{6.25}$$

By substituting equations (6.24) and (6.25) to (6.23), the (6.26) is obtained.

$$\sum_{n=1}^{N}\left[C_n \sin\left(\frac{2n\pi}{T}t\right)\right] - \frac{Z_h}{g}\sum_{n=1}^{N}\left[-C_n\left(\frac{2n\pi}{T}\right)^2 sin\left(\frac{2n\pi}{T}t\right)\right] = \sum_{n=1}^{N}\left[b_n \sin\left(\frac{2n\pi}{T}t\right)\right] \tag{6.26}$$

This equality is true when:

$$C_n - \frac{Z_h}{g}\left(-C_n\left(\frac{2n\pi}{T}\right)^2\right) = b_n \tag{6.27}$$

Solving for C_n and substituting to (6.24) we get the particular solution for the second order differential equation:

$$y(t) = \sum_{n=1}^{N}\left[\left(\frac{b_n}{\left(1 + \frac{Z_h}{g}\left(\frac{2n\pi}{T}\right)^2\right)}\right)\sin\left(\frac{2n\pi}{T}t\right)\right] \tag{6.28}$$

To get the CoM equation all we have to do is to solve the differential equation (6.23) but changing the right side of the equation with the corresponding ZMP equations (6.19) and (6.20). The results are given by equations (6.29) and (6.30).

$$CoM_x(t) = \frac{A}{T}\left(t - \frac{T}{2}\right) + \sum_{n=1}^{N}\left[\left(\frac{b_{n\_x}}{\left(1 + \frac{Z_h}{g}\left(\frac{2n\pi}{T}\right)^2\right)}\right)\sin\left(\frac{2n\pi}{T}t\right)\right] \tag{6.29}$$

$$CoM_y(t) = \frac{b}{T}\left(t - \frac{T}{2}\right) + \sum_{n=1}^{N}\left[\left(\frac{b_{n\_y}}{\left(1 + \frac{Z_h}{g}\left(\frac{2n\pi}{T}\right)^2\right)}\right)\sin\left(\frac{2n\pi}{T}t\right)\right] \tag{6.30}$$

6.3 Omnidirectional Walk Engine

Generating omnidirectional walking, consisting of straight, curved, side and diagonal walking, and being able to change the direction of the walk, improves the ability of a humanoid robot to maneuver with more agility in a dynamic environment, for example a soccer field. Recently, several researches on the locomotion of humanoid soccer robots have been published, which are mainly based on the ZMP control approaches [52][46][53]. Many of them have a brief explanation of their walk engine, but the foot planner and ZMP controller have not been explained in detail. In addition, most of them do not include an active balance method.

In this section, we present an omnidirectional walk engine to develop an autonomous soccer humanoid robot, which mainly consist of a Foot planner, a ZMP and Center of Mass (CoM) generator and an Active balance feedback loop.

In order to get a functional omnidirectional walk it is necessary to decompose it in several modules and address each module independently. Next, a description of each module is given as well as its functionalities. Figure 6.6 shows each module and how they interact with each other.

Foot Planner – Based on the given speed of the walk in X and Y direction generates future collision free support foot positions, the extended details will be given in section 6.3.1.

ZMP and CoM Trajectory Generator - In this module the ZMP trajectories is generated using the desired support foot positions and step periods. This computation takes into account only the linear component of the walk, such as forward, backward, diagonal and sidewalk. The CoM trajectory is generated based on the ZMP trajectory and the cart table model. The ZMP preview controller as a numerical approach and a proposed new Fourier based approach as an analytical approach are applied on the cart table model, which both of them can generate CoM trajectory. The analytical and numerical approach will be explained in the section 6.3.2 and 6.3.3, respectively.

Angular Velocity Controller - This module is where the angular component of the walk is controlled. It uses the angular velocity to control the orientation of both feet relative to the CoM desired orientation, which results in a controller for the CoM angular displacement. Further details will be given in section 6.3.4.

Swing Trajectory Generator - This module is responsible to generate a trajectory for the swing foot. Swing trajectory connects the current support foot to the next support foot. The cubic Bézier curve is used to generate the trajectory.

Feet Frame Computation - After computing the support foot position, CoM position and swing trajectory foot position has to be computed taking into account if it is in the double

support phase or single support phase. This module is responsible for computing the position and orientation of both feet relative to the CoM frame.

Active Balance - This module is where the balance of the humanoid during locomotion is controlled in order to maintain it stable. In section 6.3.5, a detailed explanation is presented.

Inverse Kinematics - This module is responsible to convert the positions of both feet relative to the CoM to joint angles. The method used to convert from position/orientation to joint angles is presented in [75] and [18].

Update Pose – The final step in producing omnidirectional walk engine is sending the angles computed in the Inverse Kinematics to the motor controller of each joint. Inside the NAO robot, this task will be done by its built in servo motor, and in our simulation, it will be done by our designed PID controller.

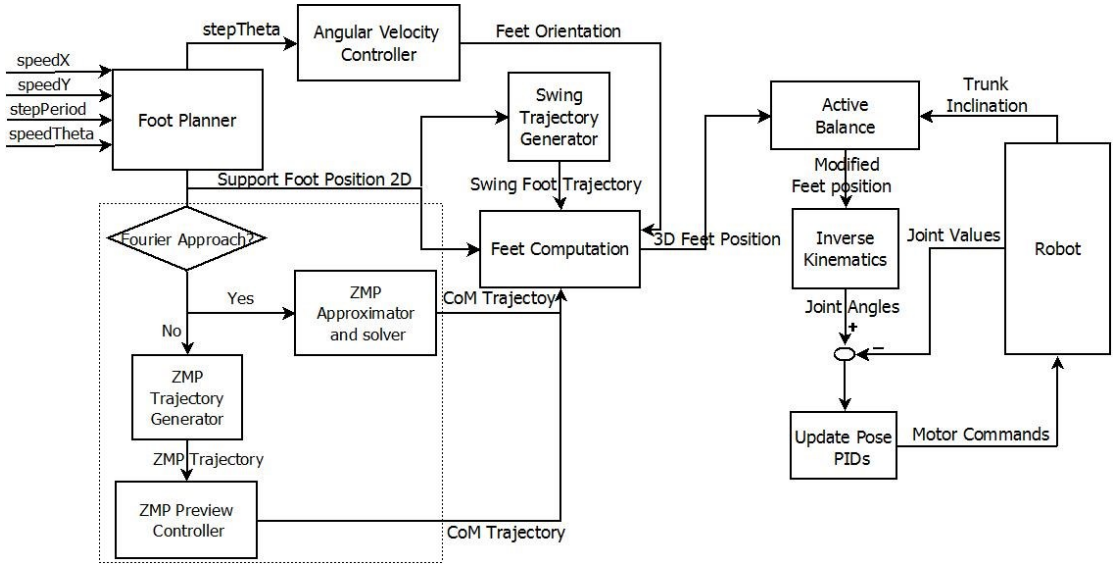

Figure 6.6: Architecture of the walk engine modules

6.3.1 Foot Planner

The future support feet positions of the robot are generated by the footstep planner. During the walking process, it is necessary to plan future steps on the basis of the current state of the feet and desired walk vector velocity. In the initial state, the robot is in double support and the Center of ZMP region (CZMP) is located at the center of the line that connects the foot centers. Then the CZMP transfers to the area of the support foot, the robot lifts its swing foot and moves it to next footprint. The CZMP changes from its initial position to a new location.

Figure 6.7 shows how we generate the next position and orientation of right (swing) foot based on velocity vector (V_x , V_y , w) in the XY plane. First, the next position of CZMP is calculated by multiplying the time duration of one step by the input velocity. This makes the linear position

and orientation change that can be used to determine the next position and orientation of CZMP *(x, y, θ)*. Then, a reference point is calculated which is used to calculate the next swing foot position. The reference point has a 55 mm distance (half distance of two legs in NAO robot), at 90 degrees, from the support foot. From the previous reference point and the CZMP position, the next reference point can be determined, from which the next support position can be derived. In order to rotate the robot θ degrees, the target footstep is rotated θ degrees relative to the CZMP frame.

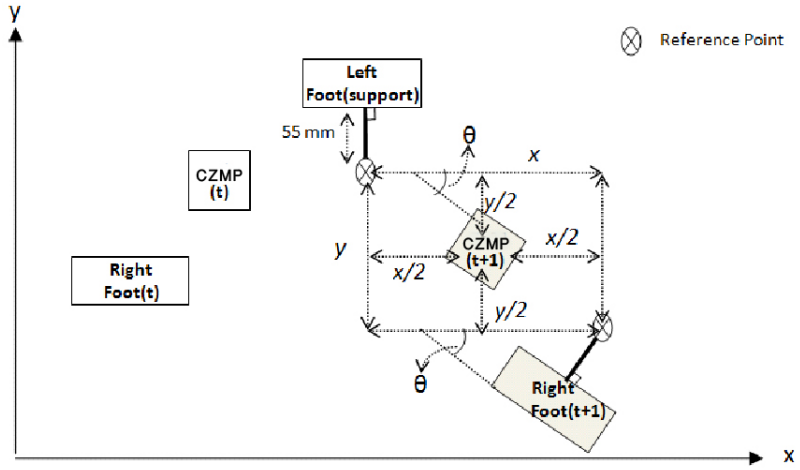

Figure 6.7: Next step Calculation of foot planner for walk vector (x, y, θ)

Two constraints are also taken into account when calculating target foot:

- Foot Reachability
- Self-Collision of feet

Foot Reachability means whether the robot is capable to move its foot to the calculated target footprint or not, and self-collision means whether the feet may collide with each other or not. These constraints are considered by defining maximum and minimum distances of feet in *x* and *y* axes, which should not be violated by new planned step and can be shown as follows:

$$A < Rightfoot.x - Leftfoot.x < B$$
$$C < Rightfoot.y - Leftfoot.y < D \quad (6.31)$$

A, B, C and *D* are constant parameters that represent the maximum and minimum relative feet positions in the global frame. The minimum step size values, A and *C*, are calculated based on the dimension of the foot. For instance, *A* and *C,* for the NAO robot are assumed to be 0.10 *m* and 0.05 *m*, respectively. The maximum value of step size, B and *D*, are determined on basis of the length of the leg and limitation of the hip joints angles. In our NAO robot experiment that will be presented in section 6.4.2, *B* and *D* are assumed to be 0.30 *m* and 0.20 *m*, respectively.

6.3.2 CoM Reference Generator using Numerical Approach

In this section, the CoM reference trajectory is modeled based on the cart-table model dynamics that was explained in 3.2.1. The CoM trajectory is generated using the ZMP preview control approach proposed by Kajita et al. [61]. The ZMP preview control approach can be categorised as a numerical approach, and an extended explanation of this approach was presented by Park et al. [67].

In the ZMP preview control approach, the jerk $\dddot{x}$ of the cart areas of the system is assumed as input u of the cart table dynamics ($\frac{d\ddot{x}}{dt}=u$). Considering this assumption, the cart table dynamics equations, which was presented in (3.2) and (3.3), can be converted to a strongly appropriate dynamical system that is presented in equation (6.32), where P is the position of the ZMP and $X = (x, \dot{x}, \ddot{x})$ is the state of CoM.

$$\begin{aligned} \frac{d}{dt}\begin{pmatrix} x \\ \dot{x} \\ \ddot{x} \end{pmatrix} &= \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}\begin{pmatrix} x \\ \dot{x} \\ \ddot{x} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} u \\ P &= \begin{pmatrix} 1 & 0 & -\frac{Z_h}{g} \end{pmatrix}\begin{pmatrix} x \\ \dot{x} \\ \ddot{x} \end{pmatrix} \end{aligned} \tag{6.32}$$

Using this Cart table model, a digital controller is designed which allows the system output to follow the input reference. The discretized system of the equation (6.33) is given below

$$\begin{aligned} x(k+1) &= AX(k) + Bu(k) \\ P(k) &= CX(k), \end{aligned} \tag{6.33}$$

Where

$$A = \begin{bmatrix} 1 & \Delta t & \Delta t^2/2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} \Delta t^3/6 \\ \Delta t^2/2 \\ \Delta t \end{bmatrix}, C = \begin{bmatrix} 1 & 0 & -\frac{Z_h}{g} \end{bmatrix}$$

By assuming the incremental state $\Delta X(k) = X(k) - X(k-1)$, the state is augmented as $\tilde{\mathrm{X}} = \begin{bmatrix} \mathrm{p(k)} \\ \Delta \mathrm{X(k)} \end{bmatrix}$, and consequently the equation (6.34) is rewritten as

$$\begin{aligned} \tilde{X}(k+1) &= \tilde{A}\tilde{X}(k) + \tilde{B}u(k) \\ P(k) &= \tilde{C}\tilde{X}(k) \end{aligned} \tag{6.34}$$

Where

$$\tilde{A} = \begin{bmatrix} 1 & CA \\ 0 & A \end{bmatrix}, \tilde{B} = \begin{bmatrix} CB \\ B \end{bmatrix}, \tilde{C} = [1 \quad 0 \quad 0 \quad 0]$$

Calculation of digital controller output is presented in equation (6.35). Figure 6.8 shows the block diagram of the system.

$$u(k) = -G_i \sum_{i=0}^{k} e(i) - G_x x(k) \tag{6.35}$$

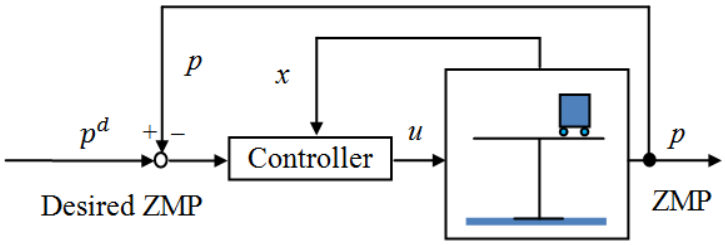

Figure 6.8: ZMP Preview control Diagram

Here, G_i and G_x are assumed as the gain for the ZMP tracking error and the gain for state feedback respectively. The ZMP tracking error is $e(i) = P - P^d$ and k denotes the k^{th} sample time. It was reported that the controller of (6.35) was not able to follow the reference ZMP sufficiently. The main cause of this issue is the inherent phase delay. For addressing this problem, the original digital controller is redesigned in equation (6.36).

$$u(k) = -G_i \sum_{i=1}^{k} e(i) - G_x x(k) - \sum_{j=1}^{NL} G_p P^d(k+j) \tag{6.36}$$

The third term consists of the planned ZMP reference up to *NL* samples in the future. Since this controller uses future information, it is called a preview controller and the gain $G_p(i)$ is called the preview gain. Our experience shows that one second of future desired walking ZMP trajectory is sufficient for the Preview controller to generate a smooth trajectory, therefore, parameter *NL* can be calculated based on the incremental time step, $NL = \frac{1}{\Delta t}$.

A detailed explanation of the preview control approach as an optimal controller technique can be found in [161]. The optimal gain, G_i, G_x, are calculated by solving discrete algebraic Riccati equation.

$$\tilde{P} = \widetilde{A^T}\tilde{P}\tilde{A} - \widetilde{A^T}\tilde{P}\tilde{B}(R + \widetilde{B^T}\tilde{P}\tilde{B})^{-1}\widetilde{B^T}\tilde{P}\tilde{A} + \tilde{Q} \tag{6.37}$$

Where $\tilde{Q} = diag\{Q_e, Q_x\}$. Then, the optimal gain is defined by

$$\tilde{G} = (R + \widetilde{B^T}\tilde{P}\tilde{B})^{-1}\widetilde{B^T}\tilde{P}\tilde{A} = [G_i \quad G_x] \tag{6.38}$$

Where $R = 1 \times 10^{-6}$. Subsequently, the optimal preview gain is recursively computed as follows:

Considering $\widetilde{A_c} = \tilde{A} - \tilde{B}\tilde{G}$

$$G_p(i) = (R + \widetilde{B^T}\tilde{P}\tilde{B})^{-1}\widetilde{B^T}\tilde{X}(i-1) \tag{6.39}$$

$$\tilde{X}(i) = \widetilde{A_c^T}\tilde{X}(i-1)$$

Where $G(1) = -G_i\begin{bmatrix}1\\0\end{bmatrix}$, $\tilde{X}(1) = -\widetilde{A_c^T}\tilde{P}\begin{bmatrix}1\\0\end{bmatrix}$, $Q_e = 1, Q_x = 0$ are assumed. Figure 6.9 shows G_p profile towards the future. We can observe that the magnitude of the preview gain quickly decreases, thus the ZMP reference in the far future can be neglected.

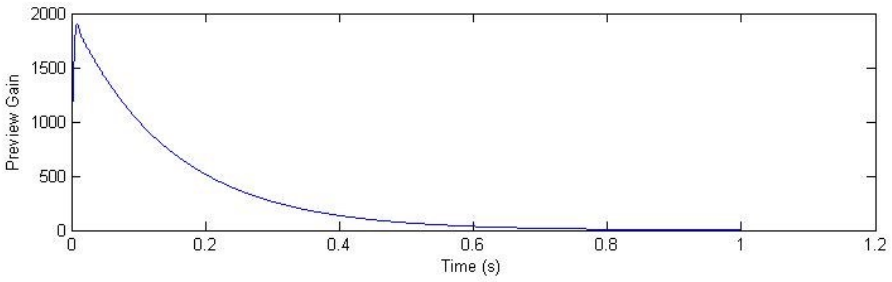

Figure 6.9: Gain calculated based on the preview controller

6.3.3 CoM Reference Generator using Analytical Approach

The CoM reference trajectory is generated based on the explained cart-table model dynamics. Our analytical approach to solve the cart-table model dynamics is explained in this section. We improve the recent general Fourier based method proposed by Park [73]. They presented the general solution of the cart-table dynamics based on the Harada's time segmentation approach [32]. Since these approaches segment the ZMP trajectories, they are called time segmentation based approaches. Park et al. have used the Harada's approach and improve it by using the Fourier series for representing the ZMP trajectory [17].

As it is generally known, a Fourier series can effectively approximate the various types of trajectories with periodic and continuous functions [73]. Fourier series can generate more smooth trajectories compared to those which are generated using Spline approximation. Using Fourier approximation is also useful for analyzing the ZMP trajectory of human walking in the frequency domain [70]. The Fourier based and Spline based time segmentation approaches have already been compared by Park [73]. Park's approach is a general solution on solving the cart-table dynamics, there is no study on how to design the time segments on a ZMP trajectory in details.

In this section we improve the recent general Fourier based method proposed by Park et al. [73]. We will emphasize on designing the time segmentation of the Fourier representation of our generated ZMP trajectories in order to solve the Cart-Table model equations analytically. The intuition behind this is improving the Fourier based approach to include double support period as a parameter to the ZMP representation. Using Fourier approach can also improve the

robustness of omnidirectional walking, since Fourier approximation can connect the ZMP trajectories of single support and double support phases smoothly.

The ZMP trajectory of a walk, in our time segmentation design, is segmented in three parts. A ZMP trajectory for two steps walking on *X* axes are shown in Figure 6.10. The robot begins to step at $t=t_0$ and the first phase of the double support ends at $t=t_1$, the single support is designed in the time between t_1 and t_2. The last part is the other double support phase, which is segmented to the time between t_2 and t_3.

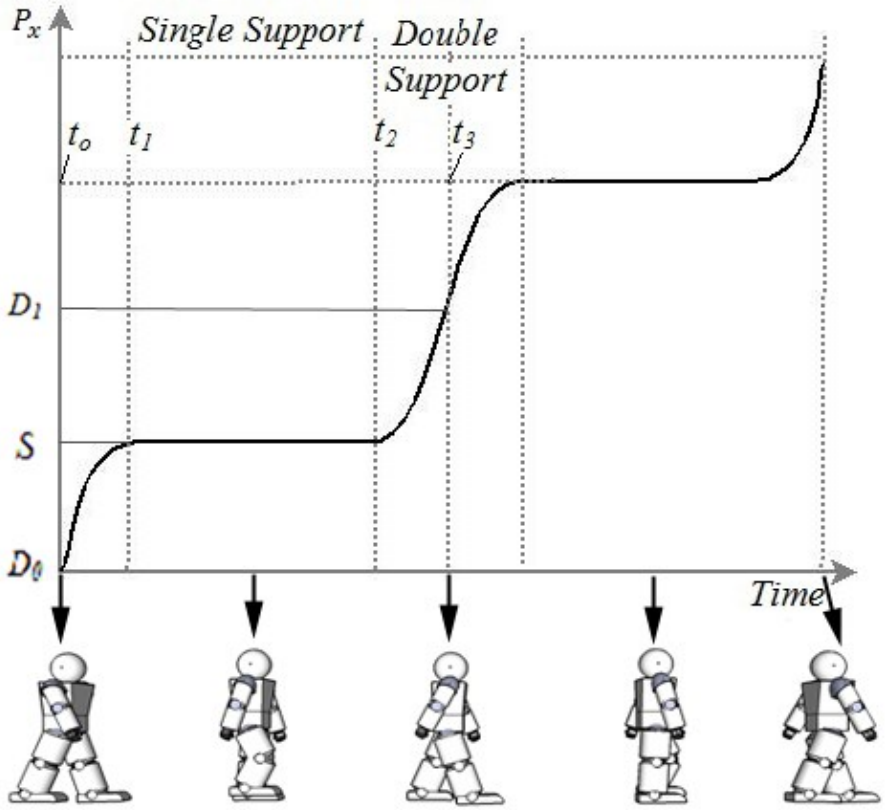

Figure 6.10: Design of the time segmentation for ZMP trajectory

Suppose that ZMP trajectory of each segment $P_j(t)$ is a periodic function and has period $2L_j$, here $L_j=t_{j+1}-t_j$, then the equation (6.40) shows the general form of the Fourier Series representation for each segment.

$$P_j(t) = {a_o}^j + \sum_{i=1}^{N} {a_i}^j \cos(i\omega_j t) + {b_i}^j \sin(i\omega_j t)$$
$$\omega_j = \frac{\pi}{L_j} \tag{6.40}$$

Where N is the order of Fourier series, j is denoting each time segments, here $j=1,2,3$, and a_o, a_i, b_i are coefficients and can be calculated by the following equations:

$$
\begin{aligned}
{a_o}^j &= \frac{1}{2L}\int_{-L}^{L} P(t)dt \\
{a_i}^j &= \frac{1}{L}\int_{-L}^{L} P(t)\cos(\frac{i\pi}{L}t)\,dt, i = 1,2,\dots,N \\
{b_i}^j &= \frac{1}{L}\int_{-L}^{L} P(t)\sin(\frac{i\pi}{L}t)\,dt, i = 1,2,\dots,N
\end{aligned}
\tag{6.41}
$$

The analytical solution of the position of the CoM can be obtained, by substituting the equation (6.40) into cart-table equation (3.2) and solving this differential equation with respect to the CoM trajectory $x(t)$. The solution is divided into homogeneous and particular parts, which are expressed in (6.42) (6.43) respectively.

$$
\begin{aligned}
x_j(t) &= {x_h}^j(t) + {x_p}^j(t) \\
{x_h}^j(t) &= V_j\cosh(T_c t) + W_j\sinh(T_c t)
\end{aligned}
\tag{6.42}
$$

$$
{x_p}^j(t) = {A_o}^j + \sum_{i=1}^{N} {A_i}^j cos(i\omega_j t) + {B_i}^j sin(i\omega_j t) \tag{6.43}
$$

Where $T_c = \sqrt{g/Z_h}$ is the natural frequency of the system and $V_j \; and \; W_j$ are the scalar coefficients, the value of them will be calculated later. Coefficients A_o, A_i, B_i can be derived from a_o, a_i, b_i. The equation (6.44) shows the Fourier representation's coefficients of the designed segments of the ZMP trajectory, shown in Figure 6.16, which is derived by basic Fourier transformation equations (6.40) (6.41):

$$
\begin{aligned}
\omega_1 &= {\pi}/{(t_1 - t_0)},\; a_o^1 = D_0 + S, \\
a_i^1 &= \frac{4\,sin^2(i\pi/2)\,(S - D_0)}{\pi^2 i^2}, b_i^1 = 0, i = 1,2,3 \\
\omega_2 &= {\pi}/{(t_2 - t_1)},\; a_o^2 = 2S, \\
a_i^2 &= 0, b_i^2 = 0, i = 1,2,3 \\
\omega_3 &= {\pi}/{(t_3 - t_2)},\; a_o^3 = D_1 + S, \\
a_i^3 &= \frac{4\,sin^2(i\pi/2)\,(S - D_1)}{\pi^2 i^2}, b_i^3 = 0, i = 1,2,3
\end{aligned}
\tag{6.44}
$$

As it is shown in Figure 6.10, D_0, S, D_1 are the starting and ending positions of the double support and single support phase. Here, the order of the Fourier series N is chosen to be 3. By choosing bigger N, usually the approximation will be more precise. In our experience by choosing the order to be 3, the approximation has an adequate precision.

By changing segmentation parameters $t_0 \ldots t_4$, the presented approach can create a walking with varied periods of double and single supports phases. Despite these mentioned advantages, the main challenge of using time segmentation method is how to keep the connectivity of the CoM trajectory between the segments. The CoM trajectory solver, equation (6.42), contains the unknown coefficients V_J, W_j, in each segment. The unknowns values should be chosen to maintain the connectivity, in order to generate the complete CoM trajectory. Similar to [32], by using equations (6.42) (6.43) and (6.44), two-point boundary value problems are designed as follows:

The initial condition of the position of CoM trajectory, where$D_0 = x_1(t_0)$, given as :

$$[1\ 0]\begin{bmatrix} V_1 \\ W_1 \end{bmatrix} = D_0 - a_o^1/2 - \sum_{i=1}^{3} a_i^1 \tag{6.45}$$

Connection of the CoM positions of two segments, on the border of each segment in t_1 and t_2. where $x_1(t_1) = x_2(t_1)$, $x_2(t_2) = x_3(t_2)$, are given as:

$$\begin{bmatrix} \cosh(T_c T_j) & \sinh(T_c T_j) & -1 & 0 \end{bmatrix}\begin{bmatrix} V_j \\ W_j \\ V_{j+1} \\ W_{j+1} \end{bmatrix} = (a_o^{j+1} - a_o^j)/2 - \sum_{i=1}^{3} a_i^j \cos(i\omega_i T_1) \tag{6.46}$$

$$, T_j = t_j - t_{j-1}, j = 1,2$$

Velocities connectivity, which are calculated based on the derivative of the equation (6.46), where:

$$\dot{x}_1(t_1) = \dot{x}_2(t_1)\ , \dot{x}_2(t_2) = \dot{x}_3(t_2) \tag{6.47}$$

The terminal condition the position of CoM, where $D_1 = x_3(t_3)$ is given as:

$$[cosh(T_c T_2)\ sinh(T_c T_2)]\begin{bmatrix} V_3 \\ W_3 \end{bmatrix} = D_1 - a_o^3/2 - \sum_{i=1}^{3} a_i^3\, cos(i\omega_i T_3) \tag{6.48}$$

By using equations (6.45) to (6.48), six unknowns V_j, W_j $(j = 1,2,3)$ can be calculated as the solution of the following linear system:

$$Y = A^{-1}B \tag{6.49}$$

Where:

$$Y = [V_1 \quad W_1 \quad V_2 \quad W_2 \quad V_3 \quad W_3]$$

$$A=\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ \cosh(T_c T_1) & \sinh(T_c T_1) & -1 & 0 & 0 & 0 \\ 0 & 0 & \cosh(T_c T_2) & \sinh(T_c T_2) & -1 & 0 \\ T_c \sinh(T_c T_2) & T_c \cosh(T_c T_2) & 0 & -T_c & 0 & 0 \\ 0 & 0 & T_c \sinh(\omega T_1) & T_c \cosh(\omega T_1) & 0 & -T_c \\ 0 & 0 & 0 & 0 & \cosh(T_c T_2) & \sinh(T_c T_2) \end{bmatrix}$$

$$T_j = t_j - t_{j-1}$$

$$B = \begin{bmatrix} D_0 - a_o^1/2 - \sum_{i=1}^{3} a_i^1 \\ (a_o^2 - a_o^1)/2 - \sum_{i=1}^{3} a_i^1 \cos(i\omega_i T_1) \\ (a_o^3 - a_o^2)/2 - \sum_{i=1}^{3} a_i^2 \\ \sum_{i=1}^{3} a_i^1 i \sin(i\omega_i T_1) \\ 0 \\ D_1 - a_o^3/2 - \sum_{i=1}^{3} a_i^3 \cos(i\omega_i T_3) \end{bmatrix}$$

In Park's approach, there is no formulation for considering the velocity connectivity at the end and the beginning of each step. Therefore, the robot can be in a bit unstable state in the beginning and end of the step. In the presented time segmentation approach, the point of changing one step to the next step is designed to be exactly in the middle of both feet. Since, the robot is in the most stable state while the robot is in the middle of the walking double support phase. This design leads to have more robust step changing.

Using Fourier approach can also improve the robustness of omnidirectional walking, since Fourier approximation can connect the ZMP trajectories of single support and double support phases smoothly. In the results section, we show that the robot could walk robustly by using the presented time segmentation design. It can also change its direction and double support period smoothly and in a real-time manner.

6.3.4 Omnidirectional Walk Rotation

In the previous subsection support foot positions and CoM trajectories were created. Using those trajectories, the robot can walk in any direction with a given linear velocity. The purpose of this section is to transform the orientation of both feet in order to allow angular displacement in order to perform curved walk and turn-in-place.

The key part to control the CoM angular displacement is the support foot, for example, to rotate the CoM orientation by θ degrees we need to rotate support foot θ degrees relative to the CoM orientation. To have an angular displacement controller we need to consider three cases:

- Right leg in the support phase:

$$LeftFoot_\theta = \frac{t}{T}d\theta \quad , RightFoot_\theta = -\frac{t}{T}d\theta$$

- Double support phase:

$$LeftFoot_\theta = k\frac{t}{T}d\theta \quad , RightFoot_\theta = k\frac{t}{T}d\theta, k = \begin{cases} -1\,, if\ d\theta \geq 0 \\ 1\,, if\ d\theta < 0 \end{cases}$$

- Left leg support phase:

$$LeftFoot_\theta = -\frac{t}{T}d\theta \quad , RightFoot_\theta = \frac{t}{T}d\theta$$

Where *t* is the time passed since the current step is starting, *T* is the current step duration and $d\theta$ is the amount of angular displacement each walk step. *LeftFoot*θ and *RightFoot*θ are the yaw angle of the end effectors on the foot related to the CoM Frame. In double support phase, when both foot are in the support phase both foot must turn in the same direction. Therefore, the variable *k* stands for generating different turn direction for both foot.

6.3.5 Active Balance Feedback Loop

After computing the support foot position, 3D CoM and swing foot trajectory, the position and orientation of both feet relative to the CoM frame should be calculated, in order to use it in the inverse kinematics module. However, keeping the balance of this generated walking cannot be guaranteed for two reasons. The first reason is the existence of some simplifications in modeling of biped walking dynamics by using the cart-table model and the other is the existence of the inherent noise in leg actuators.

An active balance approach is presented in this section, in order to reduce the risk of falling during walking, and also to adjust the trunk inclination to upright or Zero degree pitch offset. The Active balance module tries to keep an upright trunk position, by decreasing variation of trunk angles. One PD controller is designed to control the trunk angle to be Zero. An inertial measurement unit implemented in the robot body gives the trunk inclination pitch ankle. When the trunk is not upright, instead of considering a coordinate frame attached to the trunk of biped robot, position and orientation of the feet are calculated with respect to a coordinate frame, which is attached to the CoM position and tries to make the *Z* axis perpendicular to the ground plane.

The PD controller calculates the rotation angle, and the coordinate frame rotates with the calculated rotation angle. By using this transformation, the *Z* axis keeps perpendicular to the ground plane. The feet orientation is also kept constantly parallel to ground. The Transformation formulation is presented in equation (6.50).

$$Foot = T_{Foot}^{CoM}(pitchAng, rollAng) \times Foot \tag{6.50}$$

The *pitchAng* is assumed an angle calculated by the PD controller around *y* axes. Here, T_{Foot}^{CoM} is a transformation matrix in which the foot is rotating based on the *pitchAng* and *rollAng*. Figure 6.11 shows the architecture of the active balance unit, which was also on one of the modules of Figure 6.6.

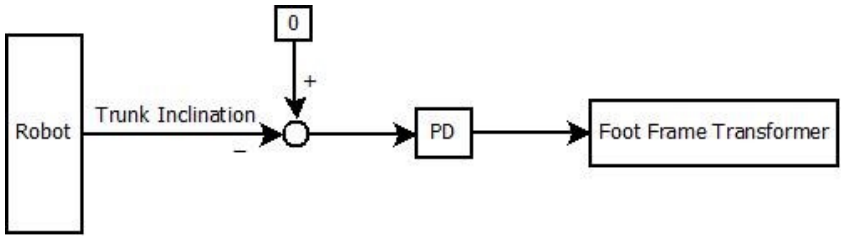

Figure 6.11: Active Balance controller diagram

By using the presented active balance approach, the *Z* axes of the base coordinate is controlled to be kept perpendicular to the ground plane. Then the foot position is calculated by using this base coordinate. Therefore, the robot tries to keep the calculated foot position on the ground or parallel to the ground even in the face of trunk inclination. The trunk inclination will be corrected and set to be zero by using the presented PD controller in Figure 6.11. Figure 6.12 shows a sequence of walking by using the active balance approach while the robot faces trunk inclination.

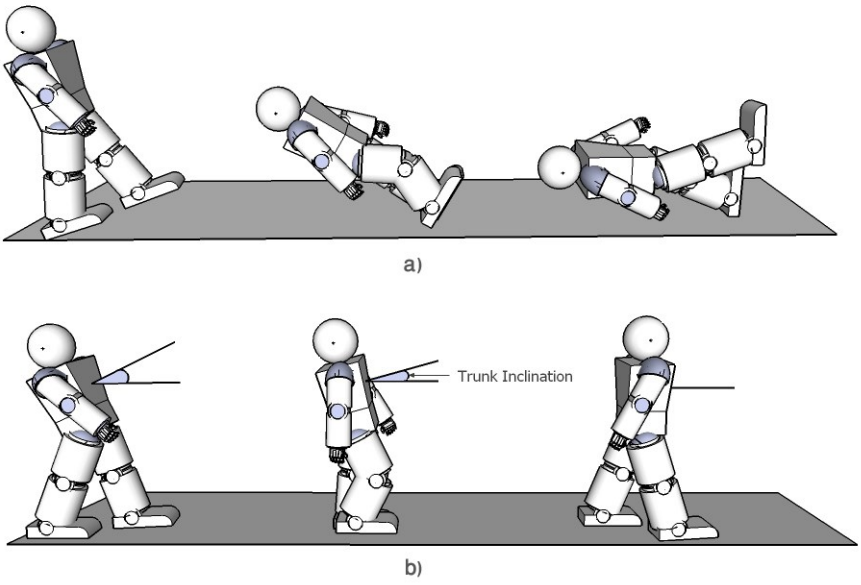

Figure 6.12: A scenario of walking while a robot face the inclination of the trunk, a) without using the active balance approach, b) using the active balance approach

6.4 Results

As it was explained in section 1.4, simulated and real NAO robots are used in this study in order to test and verify the presented bipedal locomotion approaches. The NAO robot is a kid size humanoid robot with 58 cm height and 21 degrees of freedom (DoF). As it was discussed in section 4.3, the simulation is carried out using the RoboCup soccer simulator, which is the official simulator released by the RoboCup community in order to simulate humanoids soccer match.

In this section the result of the proposed walk generation methods are given in two subsections. In the first subsection the results achieved by the single direction walk engine and our improvement on the presented Fourier approximation approach are presented. In the second subsection, the results of the proposed omnidirectional walk engine and a comparison study between our proposed analytical approach and ZMP preview control approach are given.

6.4.1Single Direction Walk Results

In order to test the performance of the proposed single-direction walk engine, a diagonal walking scenario is designed, in which the simulated NAO robot walks with 15 cm/s speed in X direction and 15 cm/s in Y direction. Parameters are used in this walking scenario are given in Table 6.2.

Table 6.2: Parameters of diagonal walking scenario

Parameters	*Value*
Step Period	0.2 s
Step Height of the swing foot	0.02 m
Step Size in X direction	6 cm
Step Size in Y direction	6 cm
Percentage of the Double Support Phase (DSP) to the whole step time	15 %
Time of whole walking	4 s, 20 steps
Height of the cart-table (Z_h)	0.22 cm

The number of the Fourier terms in equation (6.18) denoted by N is assumed equal to 3. CoM reference and CoM position projection on the ground plane for this walking scenario are shown in Figure 6.13. The generated reference CoM trajectory and executed CoM trajectory by the robot are shown in blue and red lines, respectively.

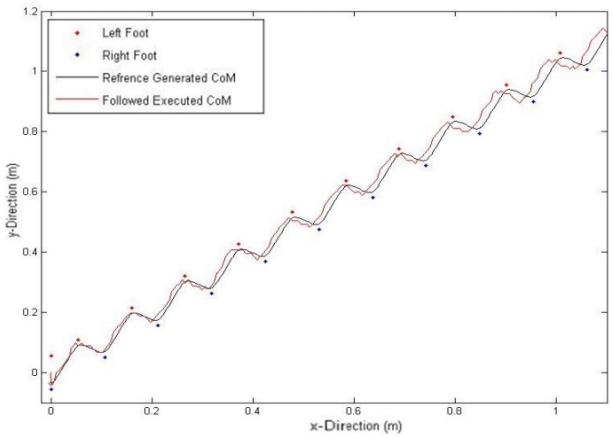

Figure 6.13: COM (lines) for the diagonal walk

For aforementioned walking scenario, the ZMP and approximated ZMP by the Fourier series and the calculated CoM are shown in Figure 6.14 and Figure 6.15.

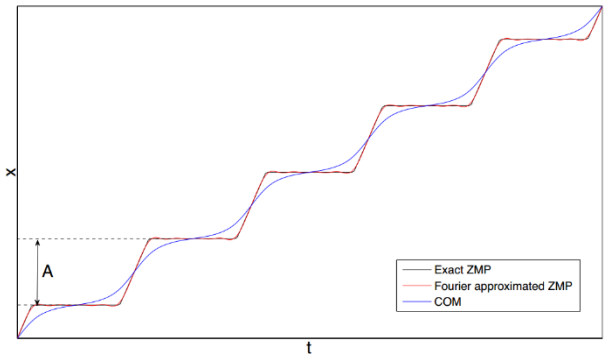

Figure 6.14: ZMP and CoM trajectory in X direction

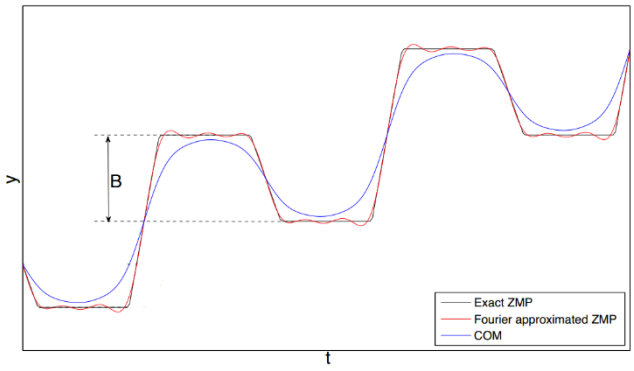

Figure 6.15: ZMP and CoM trajectory in Y direction

6.4.2 Omnidirectional Walk Results

In order to discuss and evaluate results from the omnidirectional walking engine that was presented in section 6.3, we analyze the results obtained by ZMP preview controller and presented Fourier approximation approach. The CoM trajectory is generated based on a walking

scenario that is implemented in the simulation, the NAO robot starts to walk in forward direction with the parameters presented in Table 6.3.

Table 6.3: Parameters of forward walking scenario

Parameters	*Value*
Step Period	0.4 s
Step Height of the swing foot	0.02 m
Step Size in X direction	0.12 m
Percentage of the Double Support Phase (DSP) to the whole step time	20 %
Height of the inverted pendulum (Z_h)	0.20 m

The projection of ZMP trajectory and CoM trajectory onto the ground plane on X and Y directions are shown in Figure 6.16_a and Figure 6.16_b, respectively. They have been generated by using the ZMP preview controller approach. The incremental time of the preview control loop dt is assumed to be 0.002 second. The value of the preview time parameter T is considered to be 5 second. Figure 6.16_c and Figure 6.16_d show the CoM and ZMP trajectory, which are computed by using the presented Fourier approach.

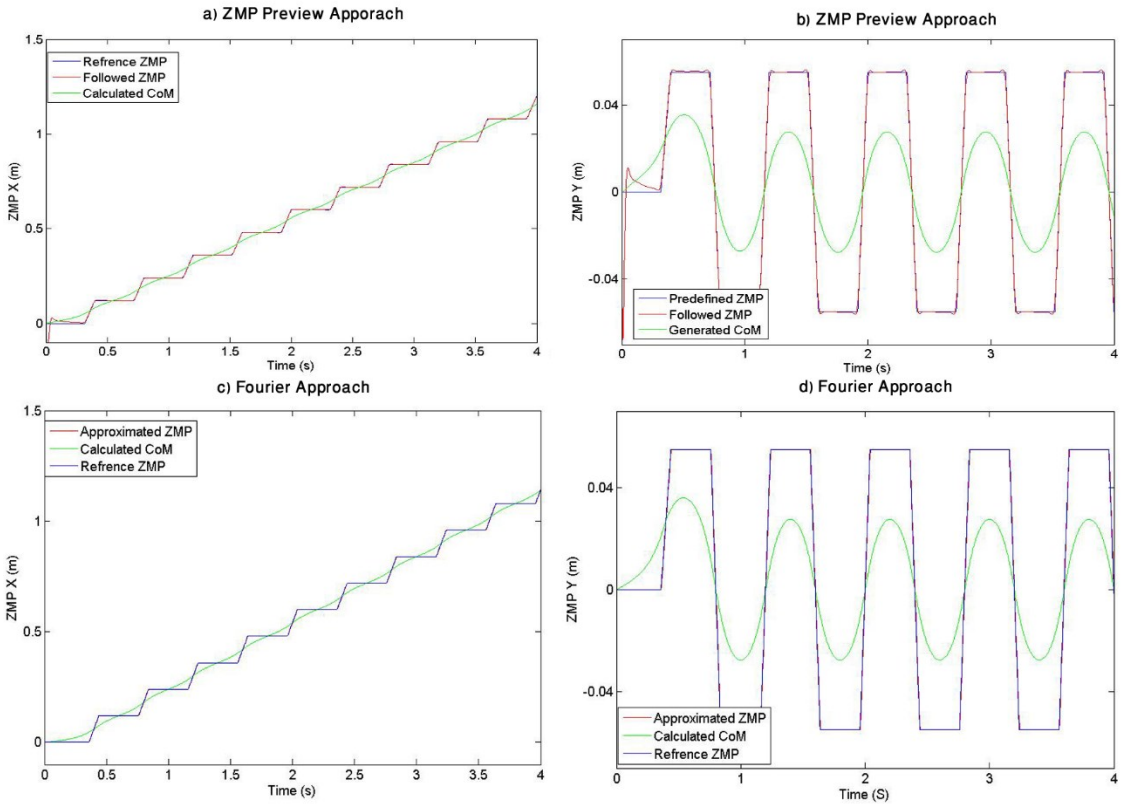

Figure 6.16: ZMP and CoM trajectories for a forward walk

In this study, the Fourier number of terms denoted by N is assumed to be 3. By choosing bigger N, usually the approximation will be more precise. In our experience by choosing the order to be 3, the approximation has an adequate precision. Based on assumed percentage of the Double

Support Phase (DSP) and walking period, the parameters of the presented time segmentation approach in section 6.3.3 are $t_0=0$, $t_1=0.08$, $t_2=0.32$ and $t_3=0.4$.

One of the drawbacks of the Fourier based approach in the literature is the lack of ability to change Double Support Phase (DSP) period during the walk. In this thesis, we have presented the approach to parameterize the double support period as the input parameter of Fourier based walk engine. In order to test this ability of our approach, we assumed the presented walking scenario in table 1 with varied DSP periods. The DSP period in the first two seconds of the walking is assumed to be 0.16 s and in the rest of the walk is considered to be 0.016 s. Figure 6.17 shows the CoM and ZMP of this walking scenario. Figure 6.18 also shows an experiment of the implemented walking scenario in the simulation.

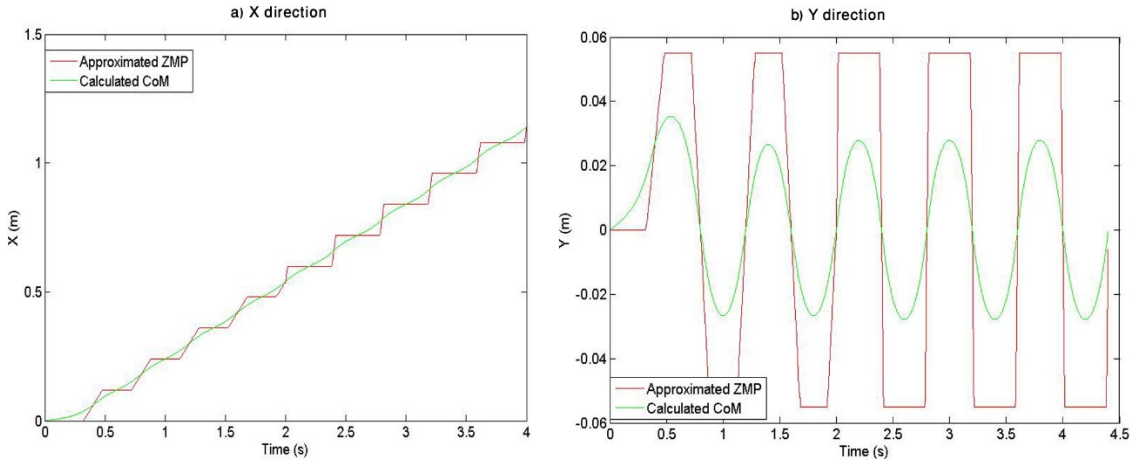

Figure 6.17: ZMP and CoM trajectory, when the DSP is changed during the walking

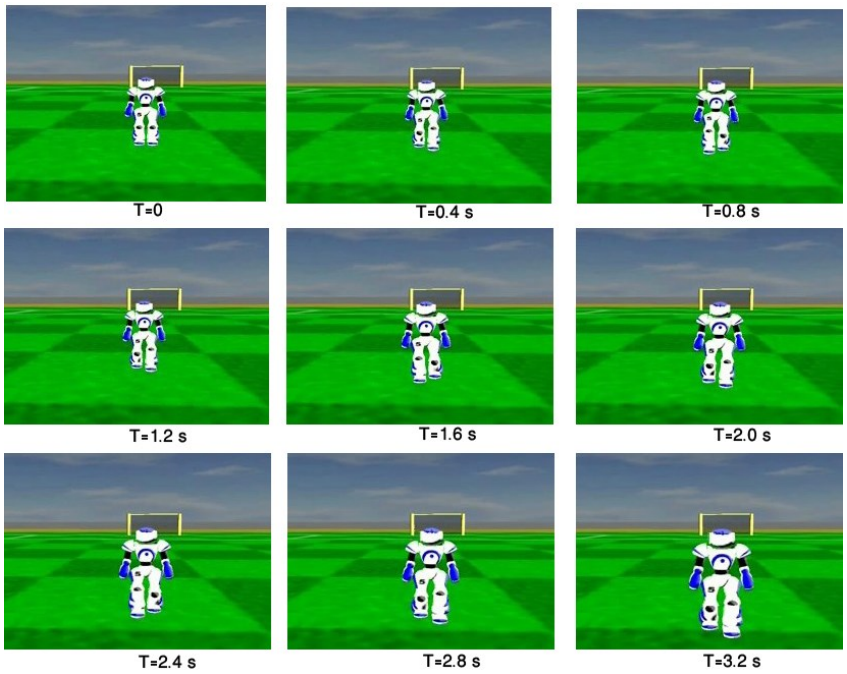

Figure 6.18: Snapshots of the proposed forward walk scenario in simulation.

As it was shown in Figure 6.16, the approximated and generated ZMP trajectories by using the presented Fourier approach and the preview controller approach are different in shape and accuracy of the approximation. As an example, in the beginning of the generated ZMP trajectory by using the preview controller, the generated ZMP trajectory could not follow the reference ZMP trajectory, since the initial ZMP tracking error is considered to be zero. In order to investigate these differences, several walking scenarios are chosen, in which the simulated NAO robot starts to walk in forward direction with different speeds for 4 seconds. The scenarios are designed based on the walk parameters in Table 6.3, but the step size is considered to be 0.04 and different step periods from 0.05s to 0.4s with 0.05s increment are assumed. All walking scenarios are simulated in the same machine with the same specification. The machine is a Pentium IV 3 GHz Core 2 Duo machine with 4 GB of physical memory. Mean Absolute Error (MAE) of the predefined reference ZMP trajectory and computed ZMP by using both Fourier and preview control approaches are presented in Table 6.4. The number of the Fourier terms denoted by N is assumed to be 3. The incremental times of the preview control loop denoted by dt are assumed to be 0.01, 0.001 and 0.0001.

Table 6.4: MAE of Computed ZMP and Reference ZMP trajectory

Step Period	*Mean Absolute Error (MAE in X direction, MAE in Y direction)*			
Second	Fourier, N=3	Preview , dt=0.01	Preview , dt=0.001	Preview, dt=0.0001
0.05	(0.000221, 0.0036)	(0.0204,0.00107)	(0.0113,0.096)	(0.00096,0.0074)
0.1	(0.000205, 0.0035)	(0.0178,0.0098)	(0.0095,0.089)	(0.00094,0.0072)
0.15	(0.000204, 0.0035)	(0.089,0.0095)	(0.0087,0.087)	(0.00078,0.0069)
0.2	(0.000198, 0.0034)	(0.0183,0.084)	(0.0066,0.086)	(0.00057,0.0064)
0.25	(0.000196, 0.0032)	(0.063, 0.0075)	(0.0058,0.086)	(0.00065,0.0061)
0.3	(0.000195, 0.0032)	(0.0046, 0.0064)	(0.0054,0.085)	(0.00032,0.0059)
0.35	(0.000195, 0.0031)	(0.00113,0.0059)	(0.0036,0.084)	(0.00027,0.0058)
0.4	(0.000194, 0.0029)	(0.0020, 0.0057)	(0.0016,0.084)	(0.00021,0.0058)

On average, the MAE in X and Y direction achieved by the proposed Fourier method are 20 and 5 times less, respectively, than achieved with the preview control approach. Although by decreasing dt, the MAE of the preview control will be decreased, achieving the performance of the Fourier approach, practically, is not possible since the execution time of the algorithm will be increased dramatically. The average and variation of execution times for different specifications on the preview control and Fourier approach are presented in the Table 6.5.

Table 6.5: The average (var) execution times of the methods

Fourier, N=3	*Preview , dt=0.01*	*Preview , dt=0.001*	*Preview , dt=0.0001*
0.0325(2.9432e-006)	0.3013(4.6517e-005)	0.8493(0.00361)	9.1097(0.0024)

After executing the methods for each walking scenarios on the machine, the average execution times for approximating ZMP and generating CoM trajectories, by using the Fourier based approach, was 0.0325 second. It is 10 times faster than the best computation time that could be achieved by using the preview control approach. For a humanoid robot that has limited computational resources, performing a real time task, such as controlling the walking, requires an algorithm with low time complexity like Fourier based approach.

In order to evaluate the performance of omnidirectional walks generated by the proposed walk engine, the robot walks in any direction and we analyze the ability of the robot to change its direction, therefore several walking scenarios were designed. The parameters of the walking engine used in the walking scenarios are the same as the previous experiments presented in Table 6.3, however the robot change its speed and direction for a couple of times.

In the first scenario, the robot must walk forward with a speed of 0.25 m/s for 10 seconds, then change to a sidewalk that lasts for 10 seconds with the speed of 0.15 cm/s, then it must do a diagonal walk for another10 seconds with the speed of -0.25 m/s and -0.15 m/s in x and y directions, respectively. At the end, the robot must stop at the starting point. In order to show the ability of the developed behavior to execute the scenario, the generated CoM trajectory and footprint projected on the ground plane are shown in Figure 6.19. Figure 6.20 also shows a generated curved walking, in which the robot walks with the speed of 0.2 m/s and rotates 6 degrees in each walk step (w=30 deg/s).

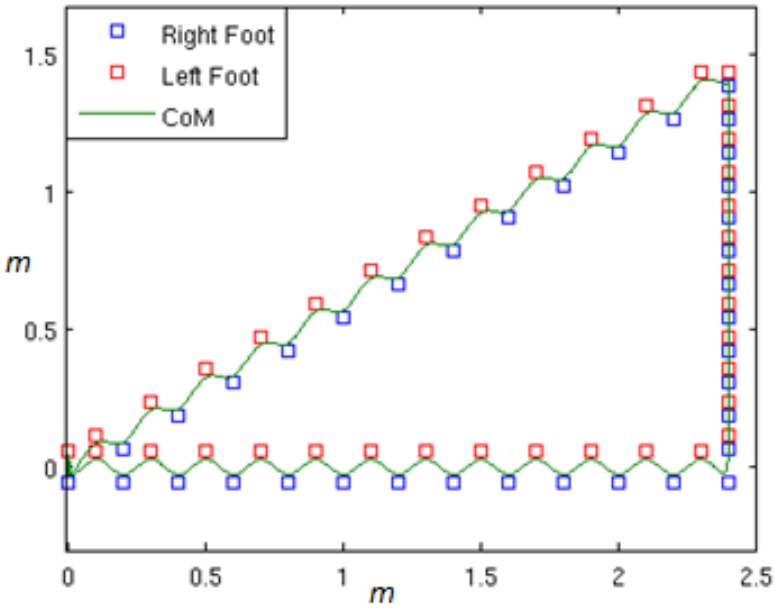

Figure 6.19: CoM and footprint of the proposed walking scenario

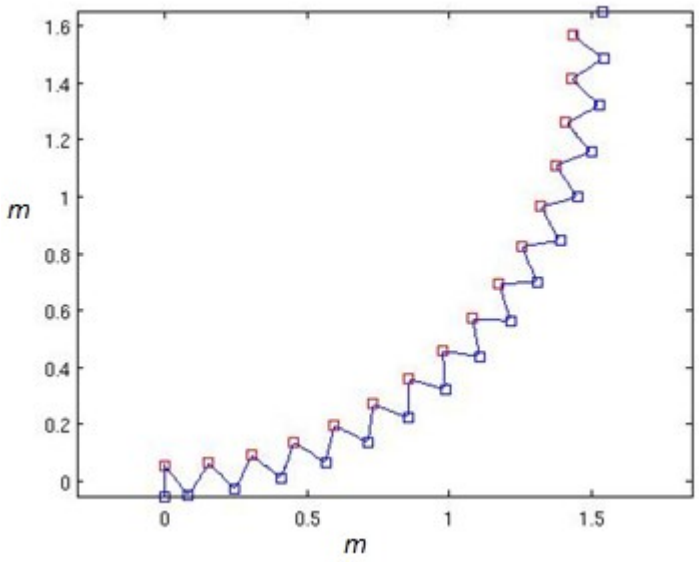

Figure 6.20: CoM and footprint of a curved walk

For testing and evaluating the active balance algorithm, the robot performs the above omnidirectional walking scenario (Figure 6.19) with two time's higher speeds. It performed this scenario by using an active balance method four times, and the robot performed it for more times without using an active balance. The *P* value of the *PD* controller of the active balance is assumed to be 0.5, and the derivative value was assumed to be 0.001. Experiments showed that in this walking scenario without using the active balance method the robot fell three times in four tests. Especially, the falls happened when the robot changed its walking direction. In contrast, by using the presented active balance method, the robot could maintain its stability in all tests; therefore, the stability of the robot in this scenario is improved 75 percent by using the active balance method.

The presented walk engine was already tested and utilized as the locomotion approach of our soccer robot team FCPortugal [16] [100] [18]. FCPortugal has been participating for several years in RoboCup soccer simulation 3D league[1], which eleven simulated NAO robots from each team play soccer with each other. Creating a good walk engine in RoboCup soccer competitions is a critical and important task for all participant teams. The walk engine should enable robots to walk stably and fast in any direction. In 2013, we changed our previous walk engine approach [18][16] to the walk engine approach presented in this chapter. Our team could achieve first place in RoboCup German open competition in 2013, and third place in 2013 international RoboCup competition[2]. One of our advantages to win in those competitions is that

[1] http://en.wikipedia.org/wiki/RoboCup_3D_Soccer_Simulation_League [Accessed: 15-Jan-2015]

[2] http://wiki.robocup.org/wiki/Soccer_Simulation_League [Accessed: 15-Jan-2015]

the robot could walk fast and change walking direction without reducing its speed, which for other teams is a very difficult task.

Table 6.6 also lists the values of some of the relevant results for the developed gait implemented on our simulated NAO compared with corresponding behavior from other teams participated in RoboCup 3D soccer simulation league. We collect times and distances/angles for forward, backward, side and diagonal (45 degrees) walk and rotate in place. To walk without rotation the agent walked for 20 seconds and recorded its position and time at each cycle. To rotate in place, the agent completed 10 full turns and recorded its time. By analyzing the results of the last RoboCup competition, the reported results are approximate, based on the average speeds of the locomotion skills for teams participating in the final round of the RoboCup 2012 competition. It is also impressive that the robot can change walking direction without reducing its speed, which for other teams is a very difficult task.

Table 6.6: Comparison between the performance of the locomotion generated by the proposed walk engine, with the skill‘s performance of other teams that participating in the final round of the RoboCup 2012 competition

Motion	*Our Agent*	*UT Austin Villa 3D*	*RoboCanes*	*Bold Hearts*	*magmaOffenburg*
Forward walk (m/s)	0.68	0.7	0.69	0.59	0.54
Backward Walk (m/s)	0.59	-	0.61	0.47	0.48
Side Walk (m/s)	0.51	-	0.47	0.42	-
Turn in Place (deg/s)	125	100	110	-	110

As it was mentioned, the required computation time and power for executing the presented Fourier based approach is less than preview control approach, therefore we used the Fourier based approach in our implementation of the walk engine on the real NAO robot. Figure 6.21, shows the sequence of snapshots of the NAO robot while it performed the presented forward walk scenarios in Table 6.3. Here, the robot could walk with a speed of around 0.25 m/s. It is the highest speed which we could achieve a stable walk by using the presented walk engine. Video of the robot performing omnidirectional walk can be found online.[3]

[3] http://paginas.fe.up.pt/~pro09025/Video/ExperimentalResults.wmv[Accessed: 15-Jan-2015]

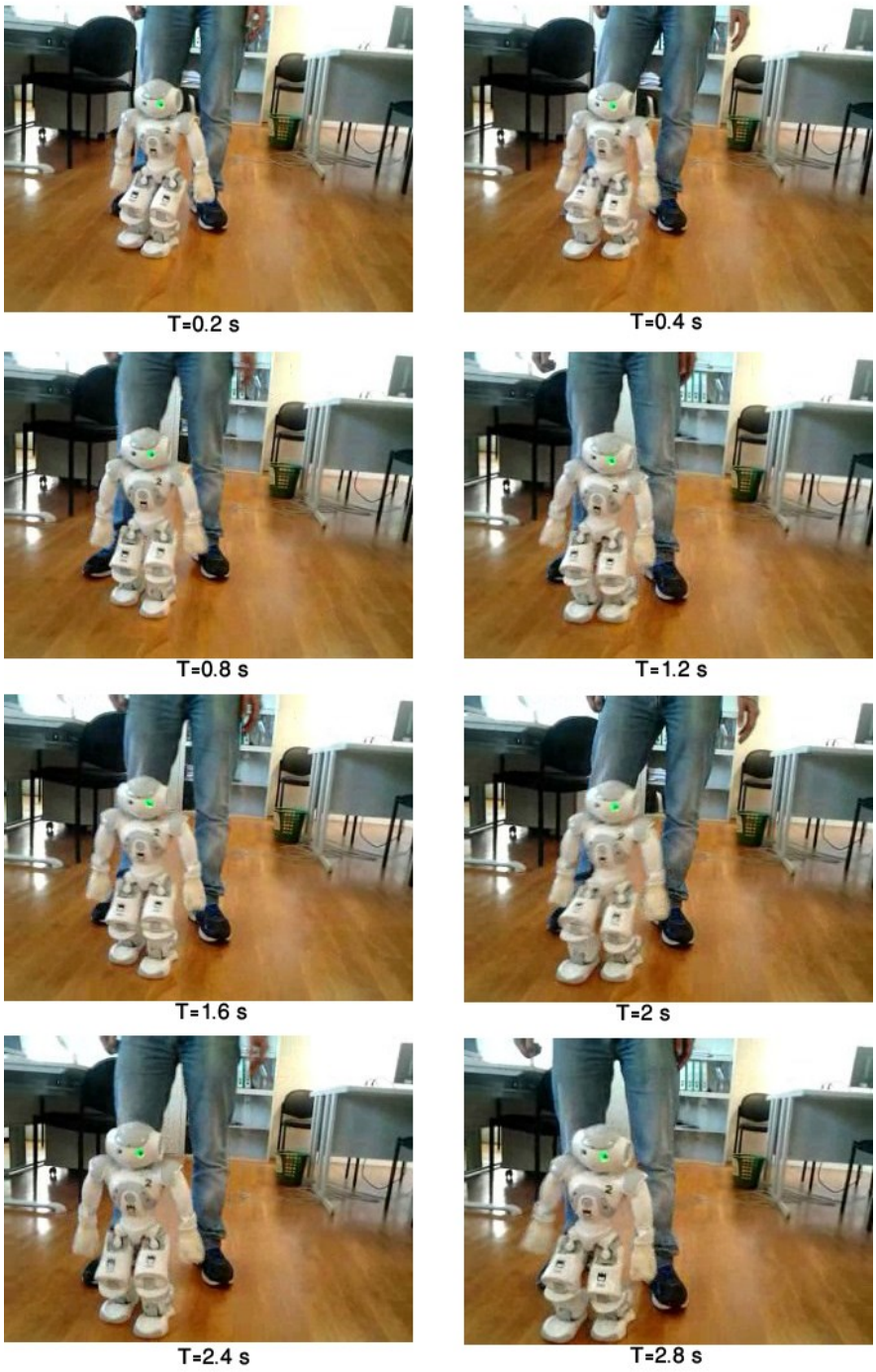

Figure 6.21: Experimental results on the NAO robot

6.5 Summary

In this chapter, we presented walking approaches using cart-table model with the focus on analytical solutions of the ZMP dynamics. This study led us to develop two different walk engines. The first one can generate walking in a single direction, and the second can generate omnidirectional walking.

In proposed single direction walk engine, an approach was developed that can generate a diagonal walk. It is for the first time that, CoM trajectories of a diagonal walk are calculated using the Fourier approximation on the predefined ZMP reference. In order to test and validate the approach, the results achieved by a simulated NAO robot performing a diagonal walking scenario were given in section 6.4.1.

The drawback of the proposed single direction walk engine is that it cannot generate stable omnidirectional walking. Since the CoM velocity is not considered as an initial condition for solving the cart-table differential equations, the CoM trajectory cannot be calculated correctly while walking direction is changed.

In order to remedy the aforementioned drawback, in section 6.3, an implementation of a ZMP based omnidirectional walk engine was presented. The preview control approach and a novel analytical approach based on Fourier approximation of ZMP were used to generate the CoM reference trajectory from the predefined ZMP trajectory. An active balance method was presented, which keeps the robot's trunk upright in case of environmental disturbances. The proposed biped locomotion approaches were tested on a simulated and a real NAO robot and used in our humanoid soccer robot team in the RoboCup competition.

Our results on generating CoM reference trajectory presented in section 6.4.2 shows that our Fourier based approach can calculate the CoM trajectory for variable double support phase periods. This improvement is achieved by using a new time segmentation-based approach. A comparison study shows that the proposed Fourier based approach as an analytical method has less time complexity and better accuracy in estimating reference ZMP trajectories than the ZMP preview approach. Using the proposed walk engine the robot performs favorably in walking fast and stable in any direction in comparison with the best RoboCup 3D soccer simulation teams. The proposed omnidirectional walk engine approach was used in FCPortugal humanoid soccer robot team, and our team could achieve first place in RoboCup German open competition in 2013, and third place in International 3D Soccer Simulation League held in Netherlands in 2013.

One of the advantages of using cart-table model is that its differential equations can be solved analytically, as it was shown in section 6.4.2 that solving analytically needs less computation process in compared to numerical approaches, which enable them for applying on humanoid robot with weak computational power.

As it was mentioned in section 3.2.1 the major drawback of cart-table model is that it assumes the CoM height fixed during walking. This simplification causes the robot to walk with bent knee and short steps. In the next chapter, we will show that this way of walking is not energy and speed efficient, and we will present an approach to remedy this drawback.

Chapter 7

Inverted Pendulum Model Applied on Humanoid Robots

In this chapter, we will investigate inverted pendulum model that is not creating bent knee. We will also use the gait learning method to optimize walking parameters with respect to speed and energy consumption. The results of this investigation led us to publish the following articles:

> **N. Shafii**, N. Lau, L. P. Reis *"Learning to Walk Fast: Optimized Hip Height Movement for Simulated and Real Humanoid Robots"* Journal of Intelligent & Robotic Systems, Springer, 2014. (IF: 0.810). (Published online)
>
> **N. Shafii**, N. Lau, L.P. Reis *"Generalized Learning to Create an Energy Efficient ZMP-Based Walking"*, RoboCup-2014: Robot Soccer World Cup XVIII, Lecture Notes in Computer Science, 2015. (In press)
>
> **N. Shafii**, N. Lau, L.P. Reis *"Learning a Fast Walking based on ZMP Control and Hip Height Movement"*, In Proceedings of the IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), pp.181-186, IEEE, 2014.

7.1 Introduction

Biped walking can be defined as the transformation of the predefined ZMP references to the possible CoM trajectory. The CoM trajectory can be determined from the ZMP using a simple physical model, approximating the bipedal robot dynamics, such as cart-table model or Linear Inverted Pendulum Model (LIPM) [63]. As it was mentioned in section 3.2.1 cart-table model and linear inverted pendulum model the ZMP dynamic by the same equations. The LIPM has been used predominantly to generate balanced humanoid walking, and it was also used in the proposed walking approach in the previous chapter. Although LIPM enables the robot to walk in any direction, it has a major drawback that is its simplification in order to make the model

linear. LIPM models the walk considering the height of Center of Mass (CoM) or hip height as a fixed constant, but biomechanical studies show that the CoM height is changing while human walking [79].

The Inverted Pendulum Model (IPM) [162] which is used in this chapter, as opposed to LIPM, can handle the motion of the CoM in Z axis. The state of the vertical CoM trajectory or hip height movement, including its position and acceleration, is the input to the IPM. Designing the vertical CoM trajectory is mainly studied in the context of biped running. Kajita et al. studied the vertical CoM trajectory that can create and modify the hopping and running motions [62]. In another study, the real-time generation of running was studied by using inverted pendulum dynamics, in which the vertical trajectory was designed simply and heuristically [163]. The study of designing vertical CoM trajectories is up to now simple and limited.

To the best of our knowledge, there is no study focused on the design of the optimal hip height trajectory generator to optimize walking speed. One of the aims of this chapter is to study the generating a fast walk with the focus on the effect of hip height movement. Our approach is based on modeling the hip height movement and learning its parameters in order to generate a fast walk.

The vertical CoM movement is also important for energy consumption [164]. A ZMP based walking with variable height can be generated by using inverted pendulum model [62]. Recently, Kormushev et al. have presented an approach for generating an energy efficient ZMP based walking [165]. They have used a vertical CoM trajectory generator, modeled by a Spline representation and a policy search reinforcement learning. This approach was applied to minimize the energy consumption of a walk with only a specific and predefined step length and step period. However, the shape of an energy efficient vertical CoM trajectory is different for each walking characteristics, including step length and step period. Another aim of this chapter is to present a new learning scenario to achieve energy efficient ZMP-based walking with variable step lengths and step periods.

In this chapter, the vertical CoM movement is considered as a periodic movement for both energy efficient walking and fast walking. The vertical CoM trajectory generator is modeled using the Fourier basis functions and programmable Central Pattern Generators (CPGs). First, the vertical CoM trajectory is generated using Fourier basis functions, then this trajectory will be the input to the CPGs based on Hopf oscillator [10]. CPGs prepare online modulation of vertical CoM trajectories in the face of changes in walking speed or direction. Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [11] is applied to optimize the CoM vertical trajectory generator parameters and walking characteristics in two learning walking objectives. One is to learn a fast walk; the other one is to learn an energy efficient walk.

In order to generate a stable walk, the position of the horizontal CoM trajectory must be calculated. First, the position of the support foot during walking is planned based on the input step length using the foot planner module, the method that is used in our foot planner module is fully explained in [12]. Then, the ZMP trajectory is designed based on the support polygon. In the next step, the horizontal CoM trajectory is obtained by solving the IPM equations. Kagami et al. presented an approach for solving the differential equations of the IPM numerically [13]. In this chapter, an improved version of Kagami's approach is presented and used to generate a more dynamic gait in which the velocity connectivity of the CoM trajectory is assured.

A trunk controller is also used to control trunk balance and inclination during walking. An initial version of this approach has been presented in [20], a similar trunk controller approach has also been presented in section 6.3.5. The trunk controller module tries to modify the trunk inclination in order to keep the trunk pitch to a target angle. The target angle of trunk pitch for generating fast walk is optimized by using CMA-ES. However, the target angle of trunk pitch for generating an energy efficient walk is assumed to be zero.

Position trajectories of the swing foot are generated using Bézier curves based on predefined footsteps. The swing foot orientation is kept parallel to the ground to reduce the effect of the contact force. Finally, legs joints angles are calculated by using inverse kinematics based on swing foot trajectory, horizontal and vertical CoM position, and support foot position. We used our modified inverse kinematic approach, which can be applied on the NAO humanoid soccer robot, see details in [18][75]. Finally, the robots' joints are controlled by simple independent PID position controllers. A system overview of the proposed methodology for learning a fast walk is provided in Figure 7.1, which illustrates the role of each component.

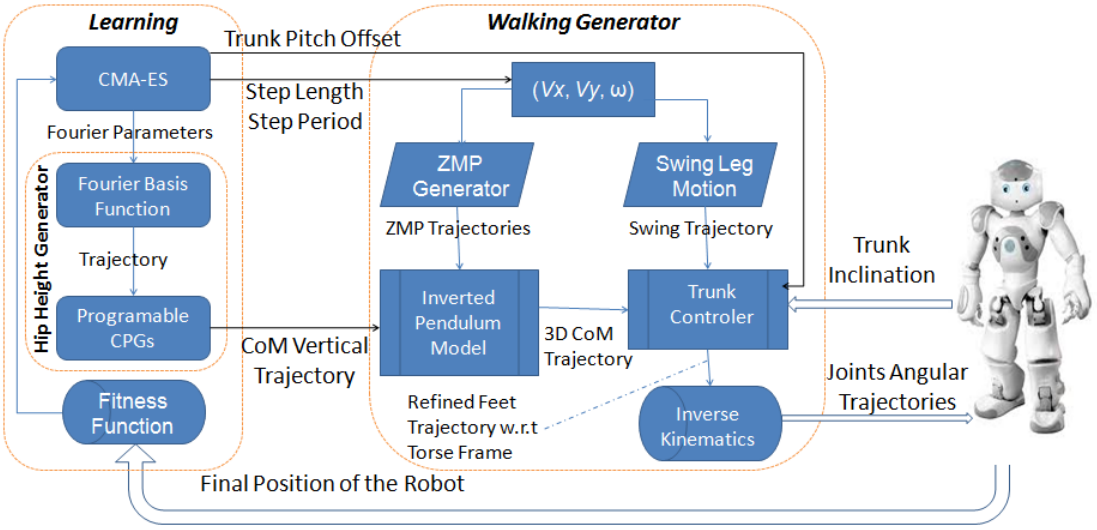

Figure 7.1: The interaction between each component of the proposed approach

In order to test its performance, the proposed learning scenario and walking generation approach is applied to both simulated and real NAO robots. The results are encouraging and show that hip height movement leads to a higher walking speed and more energy efficient walk

The remainder of the chapter is organized as follows. Section 7.2 explains the design of our model to generate vertical CoM trajectory, in which models for generating vertical CoM trajectory for both energy efficient walk and fast walk are presented. Section 7.3 presents our walking control approach including, the inverted pendulum model, our approach to generate the horizontal CoM trajectory and trunk controller approach. The learning scenarios for optimizing the walking speed and for minimizing the walking energy are explained in section 7.4. The results on both simulated and real NAO robot are given in section 7.5. General conclusion are discussed in the last section.

7.2 Vertical CoM Trajectory Generation

Many researchers, up to now, have modeled the biped walking using Linear Inverted Pendulum Model (LIPM) [63]. LIPM models a biped walk considering the hip height as a fixed constant, which creates a bent knee. Figure 7.2 shows the generated hip height trajectory in humanoid walking using the linear inverted Pendulum. Since LIPM does not allow the CoM trajectory to move vertically, the vertical CoM trajectory is shown as a straight line. It also shows the relation between the maximum step length and hip height with the maximum leg extension.

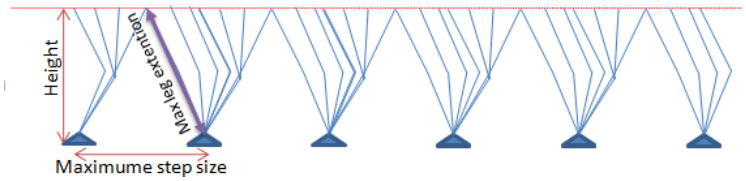

Figure 7.2: A planar view of a robot's walking while using the LIPM

As it is shown in Figure 7.2, since the maximum leg extension is a constant property, the robot should lower its hip height in order to have a bigger step size. Due to this fact, fixed height walking limits the walking step size, and leads to a lower walk speed. Biomechanical studies show that the height during human walking is variable. Figure 7.3 shows the hip height trajectory during a human walk.

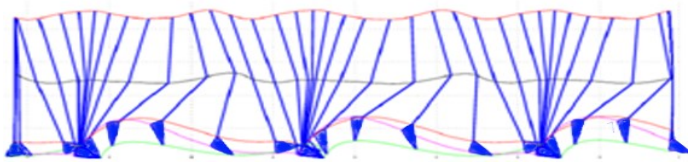

Figure 7.3: A planar view of a human walking

This chapter aims to show that the shape of the height trajectory is important to generate both fast and energy efficient walk. Designing the hip height trajectory is mainly studied in the context of the biped running [62][163]. The vertical CoM trajectory is designed simply and heuristically up to now. The study of designing vertical CoM trajectory has not been focused on generating the optimized hip height trajectory for generating fast walk and also for generating forward walk. In the following, the modeling of vertical CoM trajectory for both fast an energy efficient walk is given.

7.2.1 Vertical CoM Trajectory Model for Fast Walk

One of the contribution of this chapter is designing the hip height trajectory generator and optimizing it to generate a fast walk.

We consider the height trajectory as a periodic movement. In this study, vertical CoM trajectory is represented by Fourier basis functions. Since we know that the hip height equation is an even function, just the cosine terms are used in our hip trajectory generator. Therefore, the equation of our vertical CoM trajectory generator is given in (7.1).

$$F(t) = \beta_0 + \beta_1 \cos\left(\frac{2\pi t}{L}\right) + \beta_2 \cos\left(\frac{4\pi t}{L}\right) \tag{7.1}$$

The parameter L is equal to the step period, therefore, the generator has three parameters: β_0, β_1 and β_2. A stochastic optimization algorithm is applied in order to find the optimized hip height trajectory generator with respect to fast and stable walking. In section 4, we describe our optimization scenario. We have investigated and used similar gait representation and optimization techniques in our previous walk engine [16] [100].

The generated hip trajectory is the input to programmable CPGs. Programmable CPGs can learn any periodic input signal. When the input signal is changed, the CPGs can change the periodic signal smoothly without any sudden jerk. This is an important advantage of using CPGs in order to increase the stability of the walking in the face of switching between different walking trajectories, for example changing the direction of the walk from forward walk to the side walk. The CPGs approach used in this study is explained in section 7.2.3.

7.2.2 Vertical CoM Trajectory Model for Energy Efficient Walk

One of the contribution of this chapter is designing the hip height trajectory generator and optimizing it to generate a energy efficient walk. We consider the height trajectory as the periodic movement. In this study, vertical CoM trajectory is represented by the first five terms of the Fourier basis functions. Therefore, the equation of our vertical CoM trajectory generator is given in (7.2).

$$F(t) = \mathrm{C} + \sum_{\mathrm{i}=1}^{\mathrm{i}=2} \left(\beta_i \cos\left(\frac{i\pi t}{L}\right) + \alpha_i \sin\left(\frac{(i-1)\pi t}{L}\right) \right) \tag{7.2}$$

The parameter L is equal to the step period, therefore the generator has five parameters such as C , β_1, β_2 , α_1 and α_2. A black-box optimization approach can be applied in order to find the optimized hip height trajectory generator with respect to energy efficient walking, in the section 4 we describe our optimization scenario. The generated trajectory by Fourier basis functions is the input to the programmable CPGs.

7.2.3 Central Pattern Generator

In the CPGs implementation studies, the Hopf oscillator dynamics is interesting because of its synchronization properties when it is coupled with other oscillators or with an external drive signal. Most CPGs use phase-locking behavior for their coupling method [94]. If intrinsic frequency of the oscillator is close to the frequency component of the periodic input signal, phase-lock behavior will appear, and synchronization will be done perfectly. Righetti et al. presented a programmable CPGs approach based on Hopf oscillators which was able to learn oscillator frequency, amplitude and phase from the periodic input signals [96]. We design and implement a CPGs network, based on the programmable CPGs approach, in order to generate the hip height trajectories. The structure of our designed CPGs is shown in Figure 7.4.

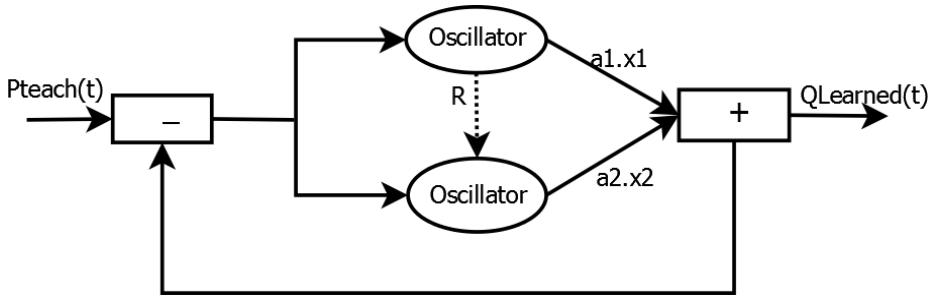

Figure 7.4: Schematic view of network of adaptive Hopf oscillators as a programable CPGs network

In this study, the external drive signal, *Pteach*, or teaching trajectory, is the trajectory that is generated by the presented Fourier based generator (7.1) in fast walk learning and (7.2) in energy efficient walk learning. Each oscillator is responsible for learning one frequency component of the signal. Since the input signal is generated by two cosine terms, the CPGs network is composed by two oscillators. The output of the system is the weighted sum of the output of the oscillators $Q_{learned}(t) = \sum_{\mathrm{i}=1}^{2} a_i x_i$, here a_i is assumed the amplitude of each learned frequency and each oscillator is denoted by i. The already learned frequencies will be subtracted from the teaching signal $F(t) = P_{teach}(t) - Q_{learned}(t)$. This negative feedback loop, leads the system to adapt to remaining frequencies components which have not yet converged. According to the fact that each oscillator has its own phase shift, a variable

encoding phase difference between the first and second oscillator is denoted by *R*. *R* contains values of the first oscillator instantaneous phase and frequency. In order to reproduce any phase relationship between the oscillators, the Kuramoto coupling scheme [90] is used. CPG's learning and dynamics are given in equations (7.3) to (7.8).

$$\dot{x}_i = \gamma(\mu - {r_i}^2)x_i - \omega_i y_i + \epsilon F(t) + \tau\, sin(\theta_i - \emptyset_i) \tag{7.3}$$

$$\dot{y}_i = \gamma(\mu - {r_i}^2)y_i - \omega_i x_i \tag{7.4}$$

$$\dot{\omega}_i = \epsilon F(t)\frac{y_i}{r_i} \tag{7.5}$$

$$\dot{a}_i = \beta x_i F(t) \tag{7.6}$$

$$\dot{\emptyset}_i = sin(\frac{\omega_i}{\omega_1}\theta_1 - \theta_i - \emptyset_i) \tag{7.7}$$

$$\theta_i = sgn(x_i)cos^{-1}(-\frac{y_i}{r_i}) \tag{7.8}$$

Equations (7.3) and (7.4), present the dynamics of each adaptive Hopf oscillator, where *x* is the generated trajectory, *i* denotes the index for each oscillator, and $r = \sqrt{x^2 + y^2}$. These oscillators have a stable limit cycle with radius $\sqrt{\mu}$ and frequency of ω. They are coupled with $F(t)$, in which γ controls the speed of recovery after perturbation, and ϵ is a coupling constant.

In equation (7.3), the Kuramoto coupling method is represented by $\tau\, sin(\theta_i - \emptyset_i)$, which is utilized to achieve phase synchronization between oscillators. The second oscillator is coupled with the first oscillator with strength τ to keep correct phase relationships between oscillators. Here, θ_i is the instantaneous phase of oscillator, and $\emptyset_i$ is the phase difference between oscillators 1 and *i*. Equation (7.5) presents the frequency adaptation with the couple input signal.

Equations (7.7) and (7.8) show how $\emptyset_i$ can converge to the phase difference between the instantaneous phase of first oscillator, θ_1 scaled at frequency ω_i and the instantaneous phase of oscillator *i*. The learning rule for updating $a_i$or amplitude is presented in equation (7.6), where β is the learning rate. This learning rule shows how correlation between x_i and *F(t)* will be maximized. The correlation will be positive on average and will stop increasing when frequency component ω_i disappears from *F(t)* because of the negative feedback loop. The learning rule is based on the negative feedback, since the input signal is linearly separable, the learning will converge. In this study the constant parameters γ, ϵ, τ and β are assumed to be 80, 400, 20 and 60 respectively. These values are obtained through trial and error method.

By applying learning rules given as differential equations, parameters such as intrinsic frequencies, amplitudes, and weights of phase coupling can be automatically adapted to a teaching signal. The main advantage of using Fourier basis functions together with the presented CPGs concept is the smoothness. CPGs generate smooth and continuous trajectories without sudden acceleration changes. By changing the direction of walk, the robot may change its optimized vertical CoM trajectory smoothly; which enables the robot not to fall during this change. CPGs also have the ability of frequency adaptation when step period and CoM trajectory period are changed.

As conclusion, Using Fourier basis functions together with presented CPG concept the proposed trajectory generator model, or policy representation, has the following advantages:

Smoothness: Using the CPGs increase basin of stability of walking. CPGs generate smooth and continuous trajectories without sudden accelerations, which enable the robot not fall and also reduce its energy consumption. By changing the step length and period during the walk, the robot may change its vertical CoM trajectory; CPGs make able this change to be smoothly. CPGs also have the ability of frequency adaptation when walking step period and CoM trajectory period changed.

Periodicity: The Fourier basis function is easily able to represent periodic or cyclic movement. A biped walk often consists of periodic movements.

Convergence: the frequency of a walk is equal to the frequency of its vertical CoM trajectory. Using Fourier basis function the frequency parameters is eliminated, as an example the robot converges to the energy efficient walk faster compared to the approaches uses Spline basis function [165].

7.3 Stable Walk Generation

Many popular approaches which have been used for controlling the balance of bipedal locomotion are based on the ZMP stability indicator. ZMP cannot generate reference walking trajectories directly, but it can indicate whether generated walking trajectories will keep the balance of a robot or not. Biped walking can be represented as a problem of balancing an IPM since in the single support phase human walking can be represented as an inverted pendulum. The robot is in balance when the position of the ZMP is inside the support polygon. When the ZMP reaches the edge of this polygon, the robot loses its balance. Biped walking trajectories can be derived from predesigned ZMP trajectory by computing a feasible CoM trajectory. The CoM trajectory can be approximated by using simplified physical models e.g., LIPM [63] and

Inverted Pendulum Model (IPM) [162]. IPM were fully explained in section 3.2.2and it will be used in this study.

In order to apply the IPM in a biped walking problem, the trajectory of ZMP positions (P_x and P_y given in (3.2) and (3.3)) during a walk must be determined. The ZMP position is assumed the position of the center of the support foot in each walk step, and then the ZMP trajectory is designed based on the ZMP positions, the step period that includes double support and single support period.

The position and acceleration of vertical CoM trajectory are the input for equations (3.2) and (3.3). Our approach to generate vertical CoM trajectory was explained in section 7.3.1. The horizontal CoM trajectory is calculated by solving the differential equations of the IPM given in (3.2) and (3.3). The main issue of using the inverted pendulum is how to solve these differential equations. The solution approach is explained in section 7.3.1. Finally, an inverse kinematics method is used to find the angular trajectories of each joint based on the planned position of the feet and generated CoM position, and the robot can walk by following the calculated angular trajectories.

7.3.1 Horizontal CoM Trajectory Generation

Kagami et al. proposed an approach to generate walking patterns by solving the ZMP equations numerically [65]. Kajita et al. used this numerical approach and the IPM in order to generate the horizontal CoM trajectory of a ZMP based biped running [62]. In this numerical approach, in order to generate horizontal CoM, first the position and acceleration of CoM are discretized with a small time-step Δt.

$$\begin{gathered} x(i\Delta t) \rightarrow x(i) \\ \ddot{x}(i\Delta t) \rightarrow \frac{x(i-1) - 2x(i) + x(i+1)}{\Delta t^2} \end{gathered} \tag{7.9}$$

Then, a tridiagonal system for the IPM differential equation given in (3.2) is written as:

$$P_x = a_i x(i-1) + b_i x(i) + c_i x(i+1) \tag{7.10}$$

Where,

$$\begin{gathered} a_i = -\frac{1}{\Delta t^2}\left(\frac{z(i\Delta t)}{g + \ddot{z}(i\Delta t)}\right) \\ b_i = 1 - 2a_i \\ c_i = -\frac{1}{\Delta t^2}\left(\frac{z(i\Delta t)}{g + \ddot{z}(i\Delta t)}\right) \end{gathered} \tag{7.11}$$

To generate the CoM trajectory, the following linear system is obtained by using (7.10).

$$\overbrace{\begin{bmatrix} P_{(1)} \\ P_{(2)} \\ \\ \\ \vdots \\ P_{(n)} \end{bmatrix}}^{ZMP_x} = \overbrace{\begin{bmatrix} b_1 & c_1 & & & \cdots & 0 \\ a_2 & b_2 & c_2 & & & \\ & a_3 & b_3 & c_3 & & \vdots \\ & & \ddots & \ddots & \ddots & \\ \vdots & & & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & & & \cdots & a_n & b_n \end{bmatrix}}^{K} \overbrace{\begin{bmatrix} x_{(1)} \\ x_{(2)} \\ x_{(3)} \\ \\ \vdots \\ x_{(n)} \end{bmatrix}}^{CoM_x} \tag{7.12}$$

In order to solve this tridiagonal system, Thomas algorithm can be applied. The solution can be obtained in *O(n)* operations. Here, $n = T_s/\Delta t$, where Δt is assumed to be 0.005 s, and T_s is the total time in which CoM is calculated.

Since *n* must be a finite value, boundary conditions must be used. Kagami et al. [65] assumed T_s to be a given time period of the walking step. They presented boundary conditions for the beginning and the end of the walking step, in which the walk was assumed to be statically stable in the beginning and end of walking step, and $c_1 = a_n = 0$. In this approach, the initial and final CoM position during a walking step need to be defined before the robot actually starts to walk [65].

There are drawbacks on this approach, including the walk in the end of each walking step is not necessarily statically balanced, and the initial velocity of the CoM has not been considered. In addition, before the robot starts to walk, it is not possible to define the exact CoM positions of the beginning and ending of the walking step.

In order to remedy the aforementioned problems, we consider the boundary conditions in equations (7.13) and (7.14).

$$P_1 = a_1 x(0) + b_1 x(1) + c_1 x(2) \tag{7.13}$$

$$P_n = a_n x(n-1) + b_1 x(n) + c_1 x(n+1) \tag{7.14}$$

We know that the initial CoM position of the current step is the last CoM position of the previous step, therefore equation (7.15) is obtained from equation (7.13) by assuming $x(0) = CoM_{initial}$.

$$P_1 - a_1 CoM_{initial} = b_1 x(1) + c_1 x(2) \tag{7.15}$$

We assume that in the last position of the CoM, the CoM velocity is zero. It means that the robot is stopped, therefore by assuming $x(n) = x(n+1)$, the following equation is given.

$$P_n = a_n x(n-1) + (b_n + c_n) x(n) \tag{7.16}$$

According to (7.15) and (7.16), the linear system given in (7.10) is rewritten by considering the new values for $P_1$and b_n , where $P_1 \leftarrow P_1 - a_1 CoM_{initial}$ and $b_n \leftarrow b_n + c_n$.

Since we know that, in a dynamic walking, the final CoM velocity is not zero, our designed controller uses future information. We plan the ZMP reference up to NL samples in the future. Our experience shows that two seconds of future desired walking ZMP trajectory is sufficient to create a dynamic walking, therefore, parameter NL can be calculated based on the incremental time step, in this case, T_s that is mentioned in equation (7.12) is assumed to be 2, and $NL = \frac{2}{\Delta t}$.

The ZMP trajectory is provided to the equation (7.10) to generate a walk that starts from the current step and last for the next two second of walking. The CoM trajectory of the current walking step is extracted from the calculated CoM trajectory. At the beginning of each walking step, the presented procedure is repeated. Last position of the CoM from the previous step is the initial CoM position for the current step.

7.3.2 Trunk Controller

After computing the support foot position, 3D CoM, and swing foot trajectory, the position and orientation of both feet relative to the CoM frame should be calculated in order to use it in the inverse kinematics module. However, keeping the balance of this generated walking cannot be guaranteed for two reasons. The first reason is the existence of some simplifications in modeling of biped walking dynamics by using the IPM, and the other is the existence of the inherent noise in leg actuators.

A trunk controller approach is presented in this section in order to reduce the risk of falling during walking, and also to adjust the trunk inclination to the desired pitch offset. This approach has already been tested and used in section 6.3.5, in which the pitch offset is assumed to be zero.

The trunk controller module tries to keep an upright trunk position by decreasing variation of trunk angles. One PD controller is designed to control the trunk angle to be the desired trunk pitch angle. An inertial measurement unit, which is included in the robot body, gives the trunk inclination angle. When the trunk angle is not the desired pitch angle, instead of considering a coordinate frame attached to the trunk of the biped robot, position and orientation of the feet are calculated with respect to a coordinate frame, which is attached to the CoM position and the Z axes always has the predefined pitch angle to the ground plane. For example, if the trunk pitch offset is assumed to be zero, the Z axes keeps always perpendicular to the ground plane.

The PD controller calculates the rotation angle based on the difference between the current trunk inclination and the desired trunk pitch angle. The calculated rotation angle is a portion of this difference, and the coordinate frame rotates with the calculated rotation angle. By using this transformation, the controller tries to keep the Z axis of the coordinate frame in a desired angle to the ground plane. The foot position is calculated by using the rotated coordinated frame. the

feet orientation also tries to be kept parallel to the ground. This Transformation formulation is presented in equation (7.17).

$$Foot = T_{Foot}^{CoM}(pitchAng) \times Foot \tag{7.17}$$

The *pitchAng* is assumed an angle calculated by the PD controller around y. Figure 7.5 shows the architecture of the trunk controller module. It was also on one of the modules of Figure 7.1.

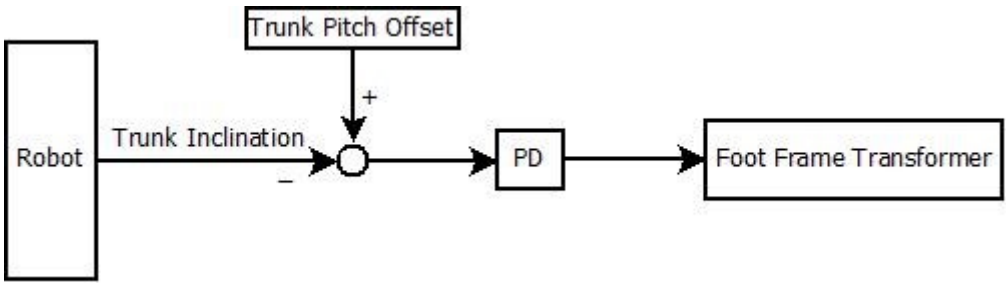

Figure 7.5: Trunk controller diagram

7.4 Learning Scenarios

In this section, Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is used as a stochastic optimization algorithm for our gait optimization scenario. CMA-ES is a population-based stochastic, derivative-free method, which can be used in black-box optimization problems. CMA-ES has been successfully applied on gait optimization scenarios [166] [135] [101], in which it has been reported that CMA-ES could achieve better results and faster convergence compared to other famous stochastic optimization techniques such as particle swarm optimization (PSO) [101] and Genetic Algorithm (GA) [135].

CMA-ES generates a set of solutions, as the population, sampled from a multivariate Gaussian distribution. After generating the population, CMA-ES evaluates each solution with respect to a fitness measure. After evaluating all generated solutions in the population, Multivariate Gaussian distribution attributes are updated based on the solutions with highest measured fitness. The Gaussian distribution mean is obtained by using the weighted average of the solutions with the highest fitness; solutions with higher fitness have higher weights. The covariance matrix of the distribution is updated such that the direction of the next generated candidates goes toward previously successful solutions.

The learning parameters and learning scenarios that are used for both learning fast walk and learning energy efficient walk will be presented in section 7.4.1 and section 7.4.2.

7.4.1 Learning Fast Walk

In order to investigate the role of hip height movement on generating a fast and stable walk, the walking parameters, such as step length, step period and trunk pitch offset, have been optimized together with parameters of the hip height trajectory in two different optimization scenarios. In the first scenario, the height is considered as a constant value. The walk parameters, that are chosen to be optimized in the first scenario, are explained in Table 7.1.

Table 7.1: Walking parameters to be optimized in learning scenario for walking with fixed height

Parameters	*Description*
L	Step length
H	Maximum Height of the Swing Foot
T	Step Period
P	Trunk Pitch Offset
β_0	Hip height trajectory offset

In the second scenario, the hip height is assumed to be changing. The trajectory is generated using equation (7.1). This means that hip height trajectory amplitudes, β_1, β_2, are optimized together with the parameters presented in Table 7.1. In both learning scenarios, the double support period is assumed to be the five percent of the whole walking step period that is denoted by *T* in Table 7.1. Double support phase of a walk is the phase in which both feet are on the ground and the ZMP moves from the previous support leg to the next support leg.

Learning scenario for the walk with variable height is performed for walking in both forward and side direction. In forward walk scenario, the step length is considered along to the *X* axis. In side walk, the step length is considered along to the *Y* axis. Learning scenarios for the forward walk with fixed height and varied height are evaluated both on simulated and real NAO robot.

Bipedal walking is known as a complicated motion since many factors affect walking style and stability such as robot's dynamics and collision between feet and the ground. In such a complex motion, relation between the hip height trajectory and walking characteristic is nonlinear. CMA-ES algorithms can be applied to find the optimized parameters values of the presented two learning scenarios with respect to generating a fast and stable walk.

Before starting CMA-ES, the size of the population and the parameters of its distribution (initial mean and covariance matrix) should be defined. The initial set of candidates is generated from the multivariate Gaussian distribution with mean $\mu = (\mu_1, \dots, \mu_n)$ and covariance matrix, $\sum$.

In this study, the CMA-ES population size is assumed to be 20. The dimension of the candidate, *n,* in the first learning scenario and in the second learning scenario is 5 and 7, respectively.

Candidates must be generated randomly from the interval [a, b]; therefore the initial mean and initial covariance matrix are as follows:

$$\mu_i = \frac{a_i + b_i}{2}, i = 1, \dots, n$$
$$\sigma_{ii} = \frac{b_i - a_i}{6}, i = 1, \dots, n \quad (7.18)$$
$$\sum = \begin{bmatrix} \sigma_{11} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_{nn} \end{bmatrix}$$

Here, *b* and *a* are the predefined upper and lower bounds, respectively. In this study, the lower and upper bound data are depicted in Table 7.2.

Table 7.2: Lower bound and Upper bound

	L (m)	*H (m)*	*T (s)*	*P (deg)*	β_0 *(m)*	β_1 *(m)*	β_2 (m)
Upper Bound	0.2	0.08	0.8	6	0.22	0.03	0.03
Lower Bound	0.01	0.04	0.05	-4	0.14	-0.03	-0.03

In order to enable the robot with a fast walking skill a fitness function based on robot's straight movement in limited action time is considered. First the robot is initialized in *x=y=0* and it walks for 12 seconds. The fitness function is calculated when the robot falls or the time duration for walking is over. The general formulation of the fitness function is expressed in equation (7.19).

For the forward walk learning, the *distanceReward* is a reward factor which is simply calculated based on the distance (meters) travelled by the robot along the *x* axis. The $DeviationPunishment$ is a punishment factor, which is calculated based on the deviation from the *x* axis or movement along the *y* axis. For the side walk learning, the fitness function formulation is the same as in (7.19), however, $DistanceReward$ and $DeviationPunishment$ are measured by the movement along *y* and *x* axis, respectively. In both side and forward walk learning, α_1 and α_2 are assumed to be 1 and 0.1. If the robot falls, a punishment factor equal to two meters is also assigned to the $FallingPunishment$ parameter.

$$Fintness = \alpha_1 DistanceReward - \alpha_2 DeviationPunishment - FallingPunishment \quad (7.19)$$

Since there is noise in robot's actuators and sensors, training walking task in this approach can be viewed as an optimization problem in a noisy and non-deterministic environment. Resampling is one of the techniques to improve the performance of stochastic optimization methods in a noisy environment, which has already been used in our gait learning approach which was presented in section 5.4. In this study, resembling factor *m* is assumed to be 3.

7.4.2 Learning Energy Efficient Walk

In this section, the vertical CoM trajectory is represented by the first five terms of the Fourier basis functions. The optimized energy efficient vertical CoM motion must be achieved for different step periods and step lengths. Since the step length and step period are continuous variables, they are discretized with a proper resolution. The boundaries of the step lengths and step period resolutions used in this work are the following:

Step Length= [0.06…..0.18] m; Resolution = 0.04

Step Period= [0.4…. 0.8] s; Resolution = 0.2

There are 12 possible combinations of the step periods and step lengths, and for each of them the energy optimization is performed. The optimized values of the Fourier basis functions terms must be found, with respect to minimization of actuator electrical power consumption. Bipedal walking is known as a complicated motion. In such a complex motion, relation between gait trajectory and walking characteristic, e.g. energy consumption, is nonlinear. CMA-ES can be applied to find the optimized parameter values of the CoM vertical trajectory generator with respect to generating an energy efficient walk.

In this learning scenario, the population size of CMA-ES is assumed to be 8. For minimization of electrical power and energy consumption, the electrical power must be measured. The electrical power for a motor can be given in a simple form by $P_m = I^2R$, where I is the current, R the resistance. The motor stall torque is calculated by $\tau = K_t I$ where K_t is the motor torque constant. By combining these expressions, the electrical power can be rewritten as $P_m = \frac{R}{K_t{}^2}\tau^2$. Therefore, in this study, the cost metric is measured by the sum of the joint-torques squared.

7.5 Results

In this section, both simulated and real NAO robots are used in order to test and verify the approaches presented in this chapter. The specification of the NAO robot was given in section 1.4. The simulation is carried out by RoboCup soccer simulator, which is the official simulator released by the RoboCup community, in order to simulate humanoids soccer match. The simulator is based on Open Dynamic Engine (ODE) and Simspark [15]. This simulator simulates different NAO robot models, we choose the standard robot model which is the most similar robot model to the real NAO robot and its link dimension is similar to the one which can be found in [2]. In this section, First results of learning of the fast walk will be presented, and then results of learning the energy efficient walk will be discussed.

7.5.1 Fast Walk Results

The walking parameters are optimized with parameters of the hip height trajectories in the two presented optimization scenarios. In the first scenario, the height is considered as a constant value. In the second scenario, the height is assumed to be variable. Using CMA-ES, the parameters are optimized with respect to the speed.

A. Learning in Simulation

In the first presented learning scenario for the forward walk with fixed hip height, running CMA-ES algorithm on a Pentium IV 3 GHz Core 2 Duo machine with 4 GB of physical memory, 2800 trials were performed in 7 hours. Finally, the robot could walk 7.8 m in 12 seconds with an average body speed of 0.66m/s and the step size of 0.136 *m*. Figure 7.6 shows the average and best fitness values during these 47 generations.

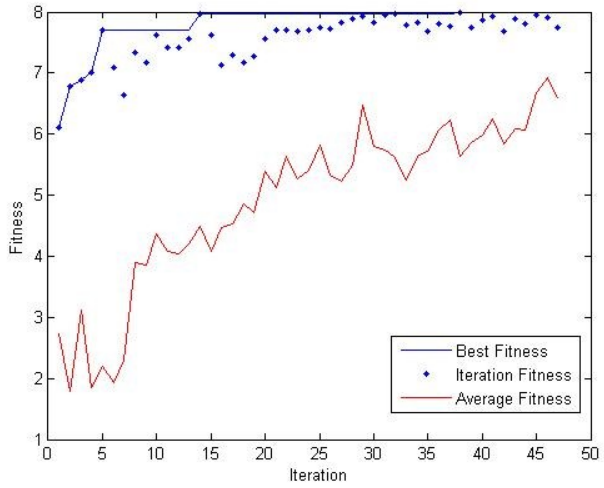

Figure 7.6: CMA-ES convergence for learning the fast forward walk with fixed height

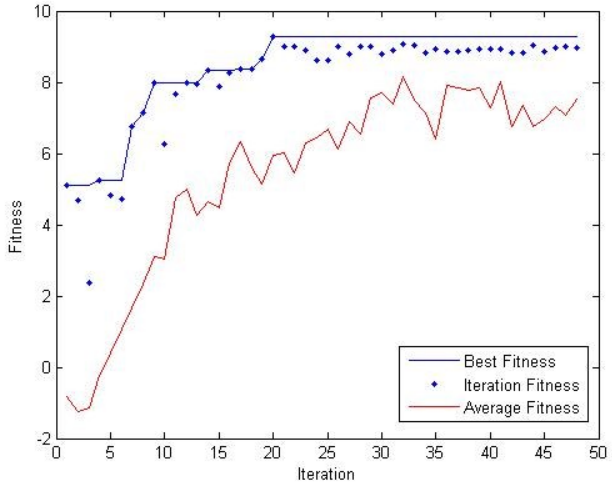

Figure 7.7: CMA-ES convergence for learning the fast forward walk with varied height

In the second learning scenario for optimizing the speed of forward walk with variable height, after 2800 trails in a machine with the same specifications, the robot could walk 9.3 *m* in 12 *s* with average body speed of 0.78 m/s and the step size of 0.155 *m*. Average and best fitness are shown in Figure 7.7 .

Table 7.3: Walking scenario parameters for the maximum speed

Parameters	*Varied height*	*Fixed Height*
Step Period	0.1983 s	0.2058 s
Swing Height	0.0751 m	0.0699 m
Step Size	0.1553 m	0.1360 m
β_0	0.1954 m	0.1773 m
β_1	0.0063 m	0
β_2	-0.0077 m	0
Trunk Pitch Offset	3.0382 deg	3.8695 deg
Achieved Speed	0.783 m/s	0.661 m/s

In simulation, the learning procedure converges to the maximum walk speed for both scenarios. The results of the presented optimization scenarios are given in Table 7.3. The achieved results show that the robot with varied hip height could walk faster with bigger step size, compared to the one with fixed hip height.

Horizontal CoM position projection on the ground plane for the fastest forward walk with varied height is shown in Figure 7.8. The aforementioned walking is started from the stop position and illustrated during the first second of walking. The optimized vertical CoM trajectory of the same walk achieved by the second learning scenario is shown in Figure 7.9 .

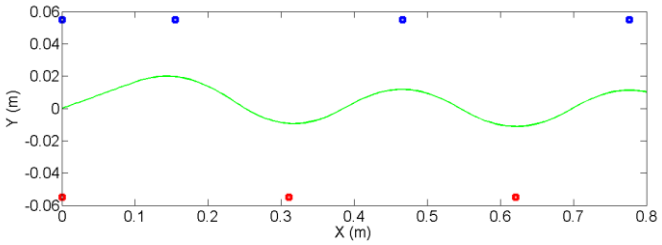

Figure 7.8: Horizontal CoM projection on the ground plane

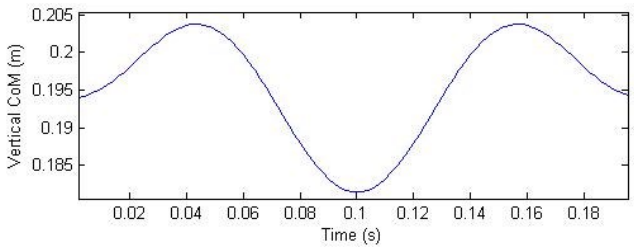

Figure 7.9: Optimized CoM vertical trajectory during one step period

For the fastest forward walk with variable height, generated ZMP trajectory and calculated CoM trajectory in the X and Y directions are shown in Figure 7.10 and Figure 7.11, respectively.

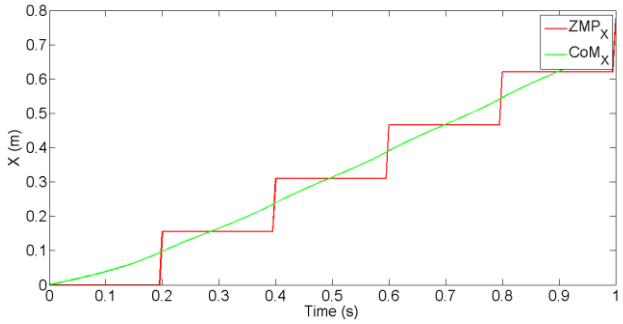

Figure 7.10: ZMP and CoM trajectory in X direction for fastest forward walk

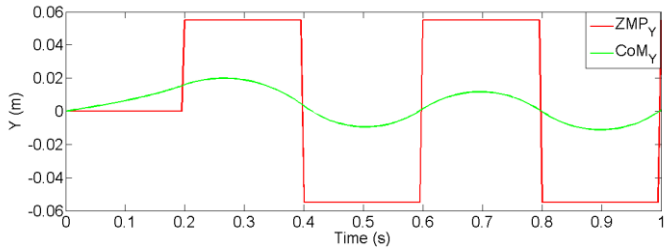

Figure 7.11: ZMP and CoM trajectory in Y direction for fastest forward walk

The side walk learning scenario is performed in simulation. The population size and resampling number are assumed to be 20 and 3, respectively. After 45 generations and 2700 iterations, the robot could walk in side direction 5.8 meters in 12 seconds, with an average speed of 0.48 m/s. The learning convergence and the learned CoM vertical trajectory are shown in Figure 7.13 and Figure 7.12 respectively. The learned values of height trajectory parameters, β_0, β_1 and β_2, are 0.187, -0.0025 and 0.0008, respectively. As it is shown in Figure 7.12, the robot learns to move its hip height slightly, about 5 *mm*. The reason behind this is due to limitation of the robot step length, and the legs cannot cross each other during a side walk. For example in this walking learning scenario, the optimized step length and period are 0.064 *m* and 0.1318 *s*, respectively.

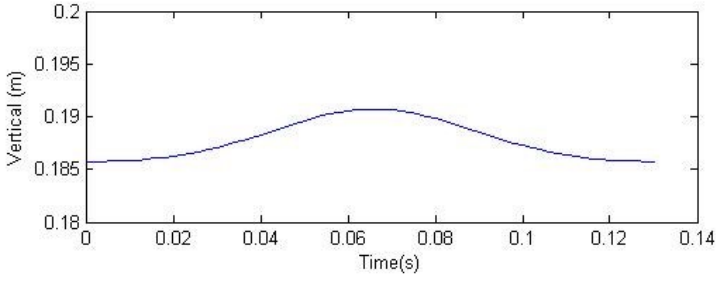

Figure 7.12: Learned vertical CoM trajectory for the side walk scenario during one step period

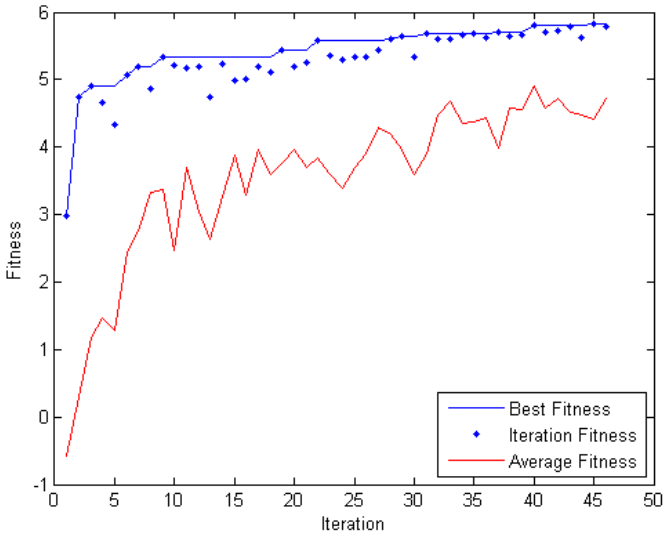

Figure 7.13: Convergence for the side walk learning scenario

Table 4 lists the achieved speeds of our learned walks with changing height compared with corresponding behavior from other teams that participated in 2013 RoboCup 3D soccer simulation league. The reported results are approximated based on the average speeds of the locomotion skills from teams participating in the final round of the RoboCup 2013 competition, taken from log-files captured from this competition[4].

Table 7.4: Comparison between performance of the locomotion generated by the proposed walk engine, with the skill's performance of other teams that participated in the final round of the RoboCup 2013 competition

Motion	*Our Agent*	*UT Austin Villa 3D*	*RoboCanes*	*Apollo 3D*	*magmaOffenburg*
Forward walk (m/s)	0.783	0.72	0.69	0.71	0.73
Side Walk (m/s)	0.48	-	0.45	0.47	-

In this competition, different types of NAO robots were used, we analyze behaviors of the standard NAO robot type that is the same type used in this study. The results show that our robot performs favourably well in walking fast and stable in both side and forward direction in comparison with the best RoboCup 3D simulation teams

As shown in Figure 7.9 and Figure 7.12, the optimized CoM vertical trajectories for different walking directions are different. By using programmable CPGs, the robot can change his

[4] http://simspark.sourceforge.net/logs/RoboCup2013 [Accessed: 15-Jan-2015]

walking speed and direction any time, and modulation of the CoM trajectories can be done autonomously**.** Figure 7.14 shows the modulation of CoM vertical trajectory of a walk when the robot changes its walking characteristics from optimized forward walk with variable height which is shown in Figure 7.9 to the optimized side walk with variable height that is shown in Figure 7.12. This change happens after about four seconds from the starting of the walk.

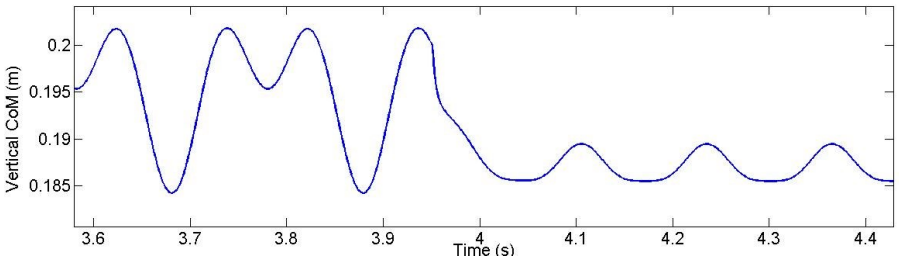

Figure 7.14: Changing vertical CoM trajectory by using CPGs based generator

For the same walking scenario shown in the Figure 7.14, we tested the change of the vertical CoM trajectories, this time with only using Fourier basis function generator, which is shown in Figure 7.15. This figure also illustrates that, at the time when the change is happening, the CoM vertical trajectory has a sudden acceleration or jerk. In our experiment, the robot in this scenario fell six times in ten tests. This happens because of the sudden acceleration in vertical CoM trajectory. Nevertheless, by using CPGs for the same walking scenario the risk of robot's falling is reduced and the robot fell two times in ten tests, because of the smooth change in vertical CoM trajectory, as it is shown in Figure 7.14.

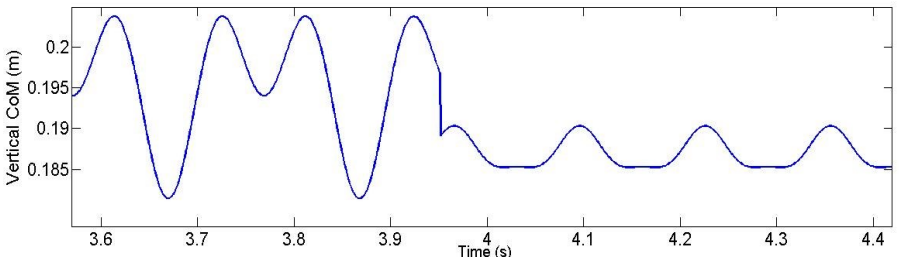

Figure 7.15: Changing the vertical CoM trajectory by the only Fourier based generator

B. Learning For Real NAO Robot

The RoboCup simulator [13] is not simulating the NAO robot accurately. We try to change the simulator to be more realistic. One of the reasons for the existing gap between the real robot and simulated one is modeling NAO's actuators. The robot's actuators in simulation can produce higher stall torque and nominal speed compared to the real robot actuators specification. Therefore, we limited the accepted actuator nominal speed and stall torque inside the simulator

to the values of real robot actuators specification which are given in [2]. Both learning scenarios presented in section 4 for optimizing forward walk are performed in this simulator, and the achieved results are evaluated and tested on a real NAO robot. Table 5 provides the learning results.

Table 7.5: Learning results to be used in Real NAO robot

Parameters	*Varied height*	*Fixed Height*
Step Period	0.249 s	0.221 s
Swing Height	0.041 m	0.052 m
Step Size	0.087 m	0.065 m
β_0	0.205 m	0.182 m
β_1	-0.001 m	0
β_2	-0.011 m	0
Trunk Pitch Offset	2.576 deg	3.839 deg
Achieved Speed	0.35 m/s	0.29 m/s

The learning convergence and the learned CoM vertical trajectory are shown in Figure 7.16 and Figure 7.17, respectively.

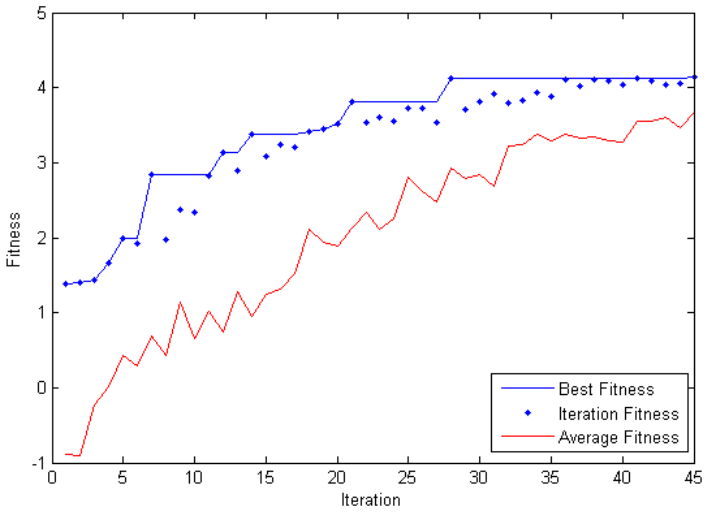

Figure 7.16: Learning convergence for the walk with variable height

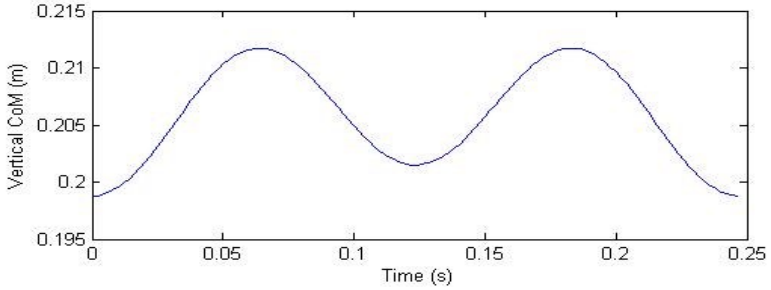

Figure 7.17: Learned vertical CoM for the Real Robot

The optimized values (see Table 7.5) which have been achieved by the walk learning with variable height are used for testing on a real NAO robot. As it was presented in Table 7.5, the robot could walk with a speed of around 0.34 m/s using variable height. It is the highest speed which we could achieve a stable walk using the presented walk engine. A video of the robot performing this walk can be found online[5]. The original walking speed of NAO prepared by Aldebaran company is about 0.119 m/s. In a recent study, the walking speed has been increased to the maximum of 0.171 m/s by using a learning approach [167]; we could achieve a higher speed compared to this study.

7.5.2 Energy Efficient Walk Results

The gait optimization for generating energy efficient walks is performed for 10 seconds walking with all the step lengths and step periods that were presented in section 7.4.2. Figure 7.18 shows the convergence of the cost function of the learning scenario for the walk with step length 0.1 *m* and step period 0.4 *s*. The optimized vertical CoM trajectory for this walk is also shown in the Figure 7.19.

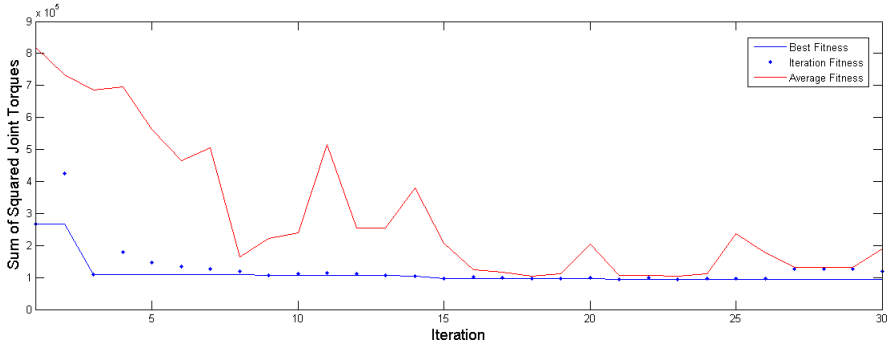

Figure 7.18: CMA_ES convergence for walk with step length 0.10 m and step period 0.4 s

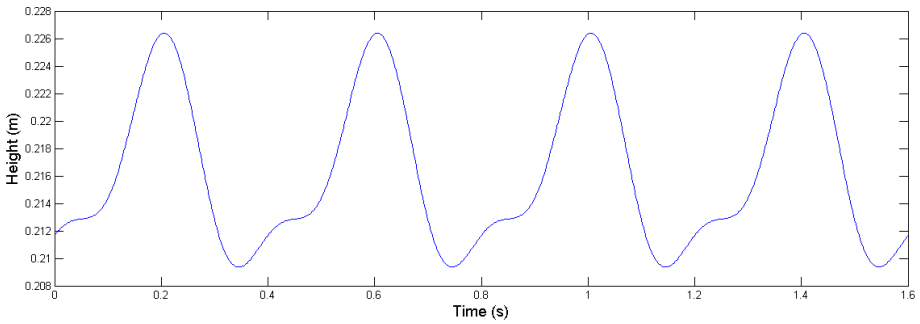

Figure 7.19: CoM vertical trajectory for a walk during 1.6 s

[5] https://www.dropbox.com/s/cyys7dtsooxevjr/Fast-Walking-Real-Robot-Experiment.wmv [Accessed: 15-Jan-2015]

The walking achieved by the above learning scenario presented in Figure 7.18 is compared to the fixed height walking with the height assumed to be the offset of learned trajectory. The results show the sum of squared torques of the walk with variable height is reduced by 25 percentage compared to the walk with fixed height. The same walking parameters with minor tuning are tested on a real NAO robot and the robot achieved to a more energy efficient walking.

The video of the fixed height walking versus the variable height walking can be found online[6]. Figure 7.20 and Figure 7.21, show the convergence of the cost function and the energy efficient vertical CoM vertical trajectory for the walk with step length 0.1 m and step period 0.8 s.

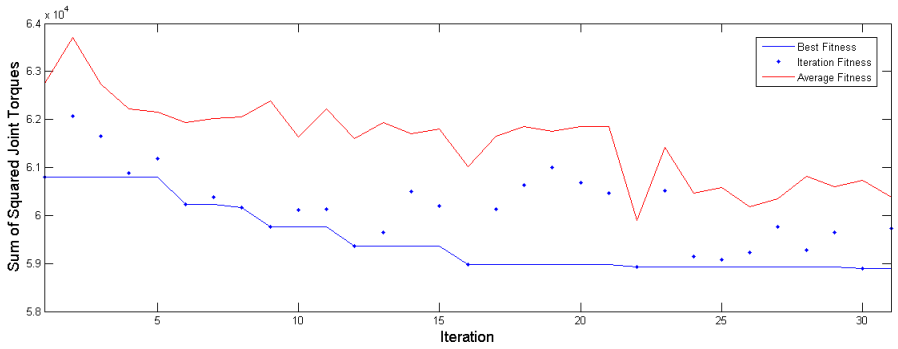

Figure 7.20: Learning convergence for walking with step length 0.10 m and period 0.8 s

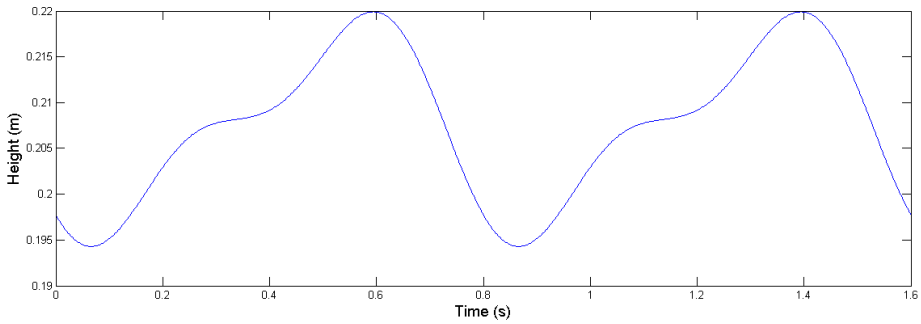

Figure 7.21: Energy efficient CoM vertical trajectory during two step periods

Figure 7.22 and Figure 7.23, also show the convergence of the cost function and the optimized vertical CoM trajectory of the walk with step length 0.14 *m* and step period 0.4 *s*.

[6] https://www.dropbox.com/s/ft1en4blotnwcrx/Real_Robot_Experiment.wmv [Accessed: 15-Jan-2015]

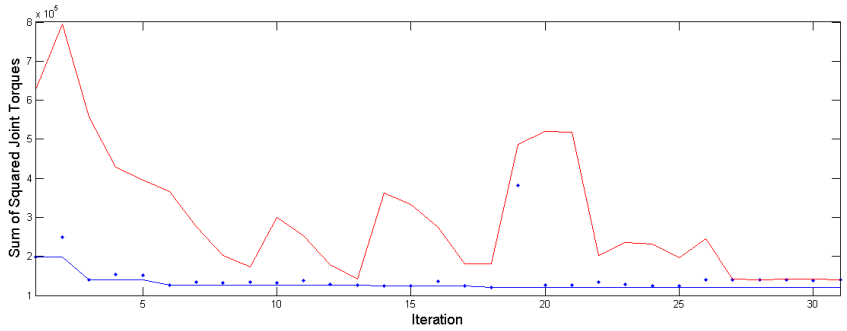

Figure 7.22: CMA_ES convergence for walk with step length 0.14 m and step period 0.4 s

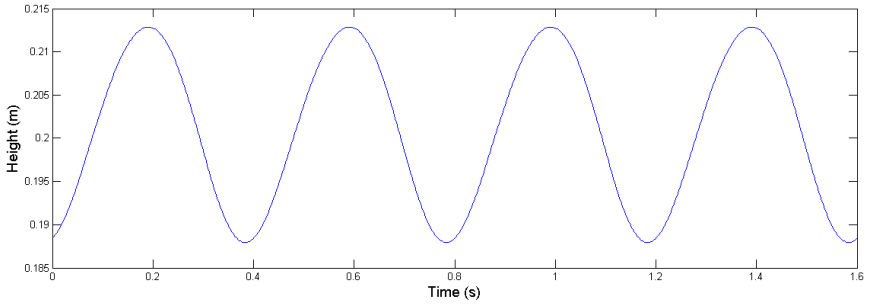

Figure 7.23:Optimized vertical CoM vertical trajectory for the above walking scenario

As shown in Figure 7.18, Figure 7.21 and Figure 7.23, the optimized CoM vertical trajectory for different walking characteristics are different. By using programmable CPGs, the robot can change its walking speed, and modulation of the CoM trajectories can be done autonomously**.** Figure 7.24 shows the modulation of CoM trajectory of a walk when the robot, changes its walking characteristics from walk with step length 0.10 *m* and step period 0.8 *s* to the walk with step length 0.14 *m* and step period 0.4. This change is happening after the four seconds from the starting of the walk.

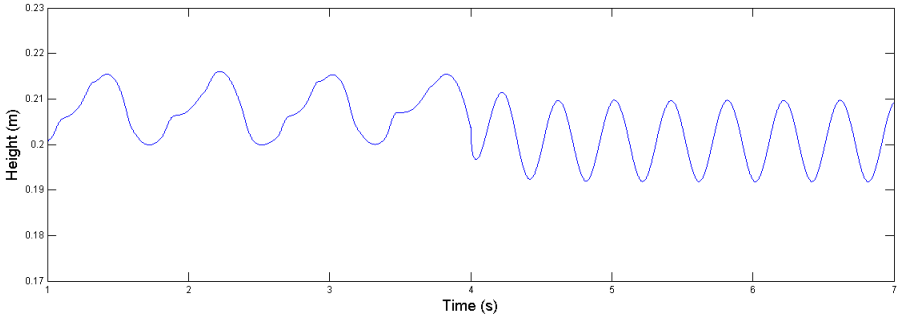

Figure 7.24: Modulation of the vertical CoM trajectory by using the CPGs

For the same walking scenario shown in the Figure 7.24, we test the change of the vertical CoM trajectories, this time with only use the Fourier basis function generator. Figure 7.25 shows this experiment. This figure also illustrates that, at the time when the change is happening, the CoM vertical trajectory has the sudden acceleration. In our experiment the robot in this scenario fell

four times in 10 tests, this happens because of the explained sudden acceleration in vertical CoM trajectory. Nevertheless, by using CPGs for the same walking scenario the robot did not fall in 10 times tests, because of the smooth change in vertical CoM trajectory, as it is shown in Figure 7.24.

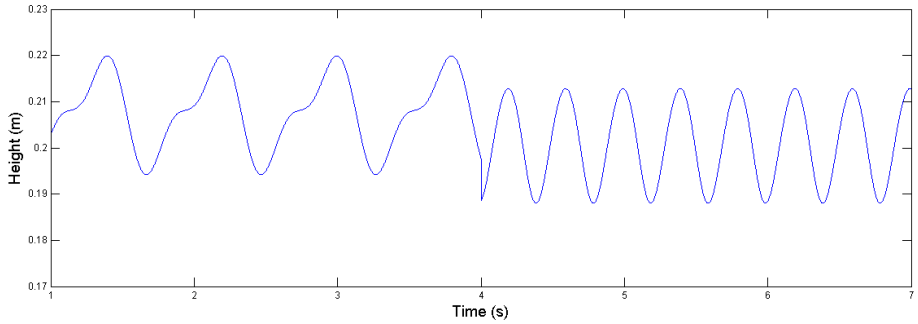

Figure 7.25: Changing the vertical CoM trajectory by the Fourier based generator

7.6 Summary

In this chapter, an Inverted Pendulum Model is used to model the ZMP dynamics. A numerical approach was presented to generate the horizontal CoM trajectory based on the predefined ZMP trajectory and vertical CoM trajectory. In this approach the walk is considered as a dynamic movement and it is not assumed to be statically stable in the beginning and end of each walking step, therefore the approach can generate a fast and dynamic walk. The intuition behind generating walking considering the CoM vertical movement is to study the hip height movement in order to generate a fast walk and also to produce an energy efficient walk. In this regards, Hip height movement is optimized for two separate learning objectives, one is for learning a fast walk, another is for learning an energy efficient walk.

For learning a fast walk, two learning scenarios were presented, and CMA-ES was used in order to optimize the hip height trajectory with respect to generating a fast walk in both forward and side directions. The results achieved by gait learning are very encouraging and show that the robot with varied height could walk faster than the robot with fixed height. Our results also show that the speed of our learned walk is higher in both side and forward direction in comparison with the best RoboCup 3D simulation teams. The learned walking parameters in the simulation were also tested on real and simulated NAO robots and we could achieve a higher speed compared to results achieved in related biped walking studies.

In energy efficient walk learning, by using the CMA-ES, an energy efficient walk is achieved for walking with different characteristics, including the step lengths and periods. The main contribution of our approach to create and energy efficient walk is the using of the CPG

approach with Fourier based function in order to formulate the vertical CoM trajectory generator. The results show that by optimizing the vertical CoM trajectory, the energy consumption of the walk is reduced by as much as 25 percent compared to the walking with fixed height. For different step lengths and periods, the optimized CoM vertical trajectory is different in shape and characteristics. By using CPGs the online modulation and change of the vertical CoM trajectories is done smoothly and without jerk.

Chapter 8

Conclusions and Future Work

This chapter gives an overview of the work reported in this thesis, referring to its main original contributions. In addition, some future work directions are also presented.

8.1 Conclusion

In this thesis, both model-based and model-free approaches and also gait optimization method were used to generate an optimized walking.

As a model-free approach, Truncated Fourier Series (TFS) approaches were improved to enable them to model walking movement in all three planes, including sagittal, coronal and transverse plane. The parameters of proposed models were optimized using gait optimization methods that were presented in Chapter 7. Using the model capable of generating walking movements in sagittal and coronal planes the robot could walk faster compared to the model that generates the walking only in the sagittal plane. Using TFS based models for generating movement in all three planes the robot could perform turn in place. The methodological part and results of these improvements were reported in Chapter 5.

Although, in this thesis, the TFS based model-free approaches were improved, we concluded that model-free approaches cannot produce a reliable and stable omnidirectional walking. According to the fact that the model has different parameters to generate different walking direction, it is not possible to find the proper parameter's values for generating all walking directions to have a fully omnidirectional walking.

In order to remedy the aforementioned problem, we decided to investigate the cart-table model, which is well-known for using in ZMP based walking approaches. In Chapter 6, an omnidirectional walk engine based on cart-table model was presented. The main difficulty of using the cart-table model is that its differential equations should be solved in order to generate CoM of the omnidirectional walking. We presented analytical approaches to solve these

differential equations using a Fourier presentation of ZMP. The analytical approaches based on Fourier series have been presented in the literature and have mainly been developed in two different directions. One can solve the differential equations of the cart-table for generating walking only in one single direction, and the other one can generate an omnidirectional walking. We improved the single directional walk approach to enable it to solve the differential equation of the cart-table model for a diagonal walking. We also improved the omnidirectional walk approach in order to calculate the CoM trajectory for variable double support phase periods. This improvement was achieved by using a new time-segmentation based approach. A comparison study showed that the proposed Fourier based approaches as analytical methods have less time complexity and better accuracy in estimating reference ZMP trajectories than the ZMP preview approach as its counterpart.

Although, by using cart-table model our robot can perform omnidirectional walk favorably compared to other RoboCup teams, this model has a general drawback that models the CoM height as a constant. This modeling can cause the robot to walk with limited walk step size and high energy consumption.

In order to remedy this problem we presented an omnidirectional walk engine using inverted pendulum model, which the CoM height can be varied during walking. The main focus of Chapter 7 was to investigate the role of hip height movement to produce high speed walking and also to generate an energy efficient walking. The hip height trajectory was modeled using similar Fourier Series (FS) modeling approach that was used in Chapter 5. In order to learn the optimized parameters of the walk and hip height model, Gait optimization methods were applied. Since the optimized hip height movement is different for the different walking directions and characteristics, Central Pattern Generators (CPG) as well-known model-free walking approachs were used. The trajectory generated by the FS was the input to the CPG. The results achieved in Chapter 7 showed that using hip height movement, the robot could walk faster and more energy efficient than using fixed hip height to walk.

As a conclusion, the first part of the thesis contribution was achieved by using model-free approach with focusing on TFS approach and gait learning methods. The second part of the contribution was achieved by applying model-based approaches, with more focus on modeling ZMP using Cart-table and Inverted Pendulum (IP) dynamics. In the last part of the contribution of Chapter 7, we used model-free based approaches including FS and CPGs to model the hip height movement and learning scenarios to achieve an optimized ZMP based omnidirectional walking, having as main goal the optimization of the energy efficiency and walk speed. Figure 8.1 shows a schematic view of the aforementioned conclusion.

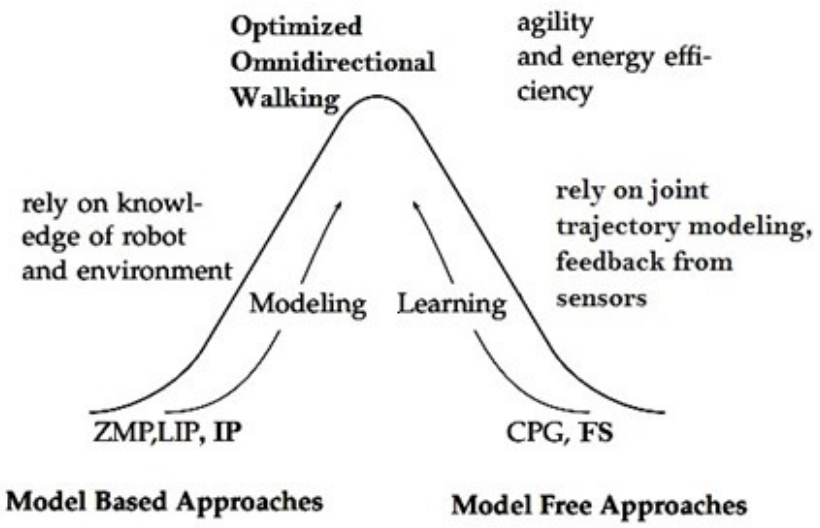

Figure 8.1: Approaches used in the thesis enabling us to achieve an optimized omnidirectional walk

All aforementioned gait generator models and biped locomotion approaches were implemented and tested both in simulated and real NAO robots. The results of the research, done in this thesis, are also being used in different humanoid research teams in RoboCup international competitions. The RoboCup simulation 3D league and the standard platform league are the two well-known soccer humanoids robot competitions, in which two teams consisting of NAO robots try to play soccer with each other. On these leagues, biped locomotion performance and agile, stable movement are very important factors that lead the robots team to win a soccer match.

Portuguese team and FC-Portugal are two research teams which have participated in the standard platform and soccer simulation leagues, respectively and benefited from the research developed on this thesis. The author is a member and active researcher of both teams since 2010. The biped locomotion approaches and results of this thesis were directly used in both robotic teams. The videos and performance of robots in the competition can be found in the following link: http://paginas.fe.up.pt/~pro09025/videos.html.

The results of the team participation in the competition are listed below:

• Rank 1st in RoboCup European Open/German Open competition 2014, soccer simulation 3D league (Humanoid Simulation), Magdeburg, Germany, April 2014.

• Rank 1st in RoboCup Portuguese Open competition 2014, soccer simulation 3D league (Humanoid Simulation), Espinho, Portugal, May 2014.

• Rank 3rd in RoboCup international competition 2013, soccer simulation 3D league (Humanoid Simulation), among 25 teams all over the world, Eindhoven, Netherlands, July 2013.

• Rank 1st in RoboCup International European Open/German Open competition 2013, soccer simulation 3D league (Humanoid Simulation), Magdeburg, Germany, April 2013.

• Rank 1st in RoboCup International European Open/Dutch Open competition 2012, soccer simulation 3D league (Humanoid Simulation), Eindhoven, Netherlands, April 2012.

• Rank 2nd in RoboCup International European Open/German Open competition 2011, soccer simulation 3D league (Humanoid Simulation), Magdeburg, Germany, March 2011.

• Rank 3rd in RoboCup International European Open/German Open competition 2010, soccer simulation 3D league (Humanoid Simulation), Magdeburg, Germany, April 2010.

8.2 Future Work

The perspective future work is related with improving the three important fundamental parts that have been used in this thesis, namely, humanoid simulation, dynamic modeling and control of the balanced robot walking and gait optimization.

The humanoid simulators have an important role in the study of this thesis. However, there is still a gap between simulation results and reality results. An ODE based simulator was used in this study. However other physical engines, recently developed, were reported more accurate than ODE, such as Bullet. It would be interesting to investigate more on humanoid simulation implementation using Gazebo or bullet that Featherstone articulated system algorithm is used [146].

In this thesis, the dynamic of a humanoid robot is modeled using simplified physical models. Until recent years, limitations on control approaches of whole-body dynamic led them to be categorized as almost purely theoretical research. Recently the state of the art of the modeling of whole body dynamics and control them have been improved which enables them to be applied in real robot walking. Preview control approach, that has been already been used in this thesis, and model predictive control approach are the techniques that recently have been used to control whole-body dynamic. So, it may be interesting to investigate on whole-body control approaches to generate dynamically stable and full-body multi-contact motions.

In this thesis, black-box optimization methods were used in gait optimization procedures. One of the drawbacks of these methods used in gait optimization is that they are not very fast and need many optimization iterations. In this thesis, all of the presented optimization scenarios were applied in simulation since the real humanoid robot cannot perform many iterations. One of the possible research directions to improve gait optimization approaches is to use more efficient black box optimizers.

The black-box optimization methods, which have been used in gait optimization procedures, are very similar to the direct policy search reinforcement algorithms that recently have been developed and used. Some new advanced reinforcement policy search methods; such as Bayesian learning, and PI^2 algorithm are attractive alternatives that can be applied to this topic.

Bibliography

[1] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka, "The development of Honda humanoid robot," in *IEEE International Conference on Robotics and Automation*, 1998, pp. 1321 – 1326.

[2] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier, "Mechatronic design of NAO humanoid," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2009, pp. 769–774.

[3] K. Kaneko, K. Harada, F. Kanehiro, G. Miyamori, and K. Akachi, "Humanoid robot HRP-3," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, 2008, pp. 2471–2478.

[4] M. Dekker, "Zero-moment point method for stable biped walking," Eindhoven University of Technology, 2009.

[5] R. C. Schafer, *Clinical biomechanics: musculoskeletal actions and reactions*. Williams & Wilkins, 1987, p. 808.

[6] H. Lim and A. Takanishi, "Biped walking robots created at Waseda University: WL and WABIAN family," *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.*, vol. 365, no. 1850, pp. 49–64, 2007.

[7] M. D. Sockol, D. A. Raichlen, and H. Pontzer, "Chimpanzee locomotor energetics and the origin of human bipedalism," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 104, pp. 12265–12269, 2007.

[8] R. Tedrake, "Underactuated Robotics: Learning, Planning, and Control for Efficient and Agile Machines: Course Notes for MIT 6.832," 2009.

[9] H. K. Khalil, *Nonlinear Sytems*, Third Edit. Prentice Hall, 2002, p. 735.

[10] A. Jain and G. Rodriguez, "An analysis of the kinematics and dynamics of underactuated manipulators," *IEEE Trans. Robot. Autom.*, vol. 9, no. 4, pp. 411–422, 1993.

[11] A. Goswami, "Postural Stability of Biped Robots and the Foot-Rotation Indicator (FRI) Point," *Int. J. Rob. Res.*, vol. 18, no. 6, pp. 523–533, Jun. 1999.

[12] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. H. Choi, and B. Morris, *Feedback Control of Dynamic Bipedal Robot Locomotion*. Taylor & Francis, CRC Press, 2007, p. 503.

[13] C. H. Steven, M. Wisse, and A. Ruina, "A Three Dimensional Passive-Dynamic Walking Robot with Two Legs and Knees," *Int. J. Robot. Res.*, vol. 17, pp. 607–615, 2001.

[14] C. M. Chew and G. A. Pratt, "Dynamic bipedal walking assisted by learning," *Robotica*, vol. 20, no. 05, pp. 477–491, 2002.

[15] J. Boedecker and M. Asada, "SimSpark – Concepts and Application in the RoboCup 3D Soccer Simulation League," *Auton. Robots*, pp. 174–181, 2008.

[16] N. Shafii, L. P. Reis, and N. Lau, "Biped Walking using Coronal and Sagittal Movements based on Truncated Fourier Series," in *RoboCup 2010: Robot Soccer World Cup XIV, Lecture Note in Computer Science*, vol. 6556, Springer, 2011, pp. 324–335.

[17] N. Shafii, L. P. Reis, and N. Lau, "Humanoid Clock-Turning Gait Synthesis based on Fourier Series And Genetic Algorithms," in *Proceedings of the 6th Doctoral Symposium in Informatics Engineering*, 2011, pp. 117–128.

[18] E. Domingues, N. Lau, B. Pimentel, N. Shafii, L. Reis, and A. Neves, "Humanoid behaviors: from simulation to a real robot," in *Progress in Artificial Intelligence, Lecture Note in Computer Science*, vol. 7026, Springer, 2011, pp. 352–364.

[19] N. Shafii, A. Abdolmaleki, N. Lau, and L. P. Reis, "A Robust Closed-Loop Gait for Humanoid Clock-Turning," in *Proceeding of Optimality Principle and Adaptation in Humanoid Robotic Control in conjunction with the IEEE-RSJ International Conference on Intelligent Robots and Systems (IROS 2012)*, 2012, pp. 20–24.

[20] N. Shafii, A. Abdolmaleki, R. Ferreira, N. Lau, and L. P. Reis, "Omnidirectional Walking and Active Balance for Soccer Humanoid Robot," *Prog. Artif. Intell. Lect. Note Comput. Scinece Scinece*, vol. 8154, pp. 283–294, 2013.

[21] R. Ferreira, N. Shafii, N. Lau, L. P. Reis, and A. Abdolmaleki, "Diagonal Walk Reference Generator based on Fourier Approximation of ZMP Trajectory," in *Proceeding of 13th international conference on autonomous robot systems (Robotica)*, 2013, pp. 94–100.

[22] N. Shafii, N. Lua, and L. P. Reis, "Development of an Omnidirectional Walk Engine for Soccer Humanoid Robots," *Int. J. Adv. Robot. Syst.*, vol. (in press), 2015.

[23] N. Shafii, A. Abdolmaleki, R. Ferreira, N. Lau, and L. P. Reis, "Towards Fast Walking based on ZMP Control and Central Pattern Generator," in *Dynamic Walking 2013*, 2013, pp. 1–3.

[24] N. Shafii, N. Lau, and L. P. Reis, "Learning a fast walk based on ZMP control and hip height movement," in *2014 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 2014, pp. 181–186.

[25] N. Shafii, N. Lau, and L. P. Reis, "Learning to Walk Fast: Optimized Hip Height Movement for Simulated and Real Humanoid Robots," *J. Intell. Robot. Syst.*, 2015.

[26] N. Shafii, N. Lau, and L. P. Reis, "Generalized Optimization and Reinforcement Learning for Creating an Energy Efficient ZMP Walker," in *Dynamic Walking 2014*, 2014, pp. 1–2.

[27] N. Shafii, N. Lau, and L. P. Reis, "Generalized Learning to Create an Energy Efficient ZMP-Based Walking," in *RoboCup 2014: Robot Soccer World Cup, Lecture Note in Computer Science*, 2015.

[28] C. L. Shih, "Analysis of the Dynamics of a Biped Robot with Seven Degrees of Freedom," in *Proceedings 1996 IEEE International Conference on Robotics and Automation*, 1996, pp. 3008–3013.

[29] A. E. Patla, "Assessment of balance control in the elderly: major issues," *Physiother. Canada*, vol. 42, no. 2, pp. 89–97, 1990.

[30] M. Vukobratović and D. Juricić, "Contribution to the synthesis of biped gait," *IEEE Trans. Biomed. Eng.*, vol. 16, no. 1, pp. 1–6, 1969.

[31] M. Vukobratovic and B. Borovac, "Zero-moment point — thirty five years of its life," *Int. J. Humanoid Robot.*, vol. 1, no. 1, pp. 157–173, 2004.

[32] K. Harada, S. Kajita, K. Kaneko, and H. Hirukawa, "An Analytical Method on Real-time Gait Planning for a Humanoid Robot," *Int. J. Humanoid Robot.*, vol. 3, no. 01, pp. 1–19, 2006.

[33] M. Vukobratovic, D. Stokic, B. Borovac, and D. Surla, *Biped Locomotion: Dynamics, Stability, Control and Application*. Springer, 1990, p. 349.

[34] D. Kleppner and R. J. Kolenkow, *An Introduction To Mechanics*. McGraw-Hill Book Company, 1973, p. 289.

[35] S. Nakaura and M. Sampei, "Balance control analysis of humanoid robot based on ZMP feedback control," in *IEEE/RSJ International Conference on Intelligent Robots and System*, 2002, pp. 2437–2442.

[36] M. Popovic, A. Goswami, and H. Herr, "Ground reference points in legged locomotion: Definitions, biological trajectories and control implications," *Int. J. Rob. Res.*, vol. 24, no. 12, pp. 1013–1032, 2005.

[37] J. Morimoto, S. Hyon, G. Cheng, D. Bentivegna, and C. G. Atkeson, "Modulation of simple sinusoidal patterns by a coupled oscillator model for biped walking," in *IEEE International Conference on Robotics and Automation, ICRA 2006.*, 2006, pp. 1579–1584.

[38] J. Morimoto and C. G. Atkeson, "Learning Biped Locomotion: Application of Poincare-map-based reinforcement learning," *IEEE Robot. Autom. Mag.*, vol. 14, no. 2, pp. 41–51, 2007.

[39] I. R. Manchester, M. M. Tobenkin, M. Levashov, and R. Tedrake, "Regions of Attraction for Hybrid Limit Cycles of Walking Robots," *arXiv Prepr.*, pp. 1–8, 2010.

[40] H. Gritli, N. Khraief, and S. Belghith, "Chaos control in passive walking dynamics of a compass-gait model," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 18, pp. 2048–2065, 2013.

[41] T. McGeer, "Passive dynamic walking," *Int. J. Rob. Res.*, vol. 9, pp. 62–82, 1990.

[42] I. R. Manchester, U. Mettin, F. Iida, and R. Tedrake, "Stable dynamic walking over uneven terrain," *Int. J. Rob. Res.*, vol. 30, pp. 265–279, 2011.

[43] M. Wisse, G. Feliksadal, J. van Frankenhyyzen, and B. Moyer, "Passive-based walking robot," *IEEE Robot. Autom. Mag.*, vol. 14, pp. 52–62, 2007.

[44] C. Graf, H. Alexander, and R. Thomas, "A Robust Closed-Loop Gait for the Standard Platform League Humanoid," in *In Proceedings of the Fourth Workshop on Humanoid Soccer Robots in conjunction with the IEEE-RAS International Conference on Humanoid Robots*, 2009, pp. 30–37.

[45] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura, "The intelligent ASIMO: System overview and integration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002, pp. 2478–2483.

[46] D. Gouaillier, C. Collette, C. Kilner, and A. Robotics, "Omni-directional Closedloop Walk for NAO," in *IEEE-RAS International Conference on Humanoid Robots*, 2010, pp. 448–454.

[47] I.-W. Park, J.-Y. Kim, and J.-H. Oh, "Online Walking Pattern Generation and Its Application to a Biped Humanoid Robot — KHR-3 (HUBO)," *Advanced Robotics*, vol. 22. pp. 159–190, 2008.

[48] L. Yang, C.-M. Chew, T. Zielinska, and A.-N. Poo, "A uniform biped gait generator with offline optimization and online adjustable parameters," *Robotica*, vol. 25, pp. 549–565, Mar. 2007.

[49] K. Matsuoka, "Sustained oscillations generated by mutually inhibiting neurons with adaptation," *Biol. Cybern.*, vol. 52, no. 6, pp. 367–376, 1985.

[50] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka, "The Development of Honda Humanoid Robot," in *IEEE International Conference on Robotics and Automation*, 1998, pp. 1321–1326.

[51] S. Hong, Y. Oh, Y. Chang, and B. You, "An omni-directional walking pattern generation method for humanoid robots with quartic polynomials," in *in IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, pp. 4207–4213.

[52] C. Graf and T. Rofer, "A closed-loop 3D-LIPM gait for the RoboCup Standard Platform League humanoid," in *Soccer Humanoid workshop Fourth Workshop on Humanoid Soccer Robots in conjunction with IEEE-RAS International Conference on Humanoid Robotics*, 2010, pp. 15–22.

[53] J. Strom and G. Slavov, "Omnidirectional walking using zmp and preview control for the nao humanoid robot," in *RoboCup 2009: Robot Soccer World Cup XIII, Lecture Note in Computer Science*, vol. 5949, Springer, 2010, pp. 378–389.

[54] F. Xue, X. Chen, J. Liu, and D. Nardi, "Real time biped walking gait pattern generator for a real robot," in *RoboCup 2011: Robot Soccer World Cup XV, Lecture Notes in Computer Science*, vol. 7416, Springer, 2012, pp. 210–221.

[55] K. Nagasaka, Y. Kuroki, S. Suzuki, Y. Itoh, and J. Yamaguchi, "Integrated motion control for walking, jumping and running on a small bipedal entertainment robot," in *IEEE International Conference on Robotics and Automation*, 2004, pp. 3189–3194.

[56] S. Kajita, T. Nagasaki, K. Kaneko, K. Yokoi, and K. Tanie, "A Running Controller of Humanoid Biped HRP-2LR," in *IEEE International Conference on Robotics and Automation*, 2005, pp. 616–622.

[57] L. Yang, C.-M. Chew, Y. Zheng, and A.-N. Poo, "Truncated Fourier series formulation for bipedal walking balance control," *Robotica*, vol. 28, no. 01, pp. 81–96, Apr. 2009.

[58] J.-Y. Kim and Y.-S. Kim, "Whole-Body Motion Generation of Android Robot Using Motion Capture and Nonlinear Constrained Optimization," *Int. J. Humanoid Robot.*, vol. 10, no. 2, 2013.

[59] Z. Liu, L. Wang, C. L. Philip Chen, X. Zeng, Y. Zhang, and Y. Wang, "Energy-efficiency-based gait control system architecture and algorithm for biped robots," *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.*, vol. 42, pp. 926–933, 2012.

[60] Q. Huang, K. Yokoi, S. Kajita, K. Kaneko, H. Aral, N. Koyachi, and K. Tanie, "Planning walking patterns for a biped robot," *IEEE Trans. Robot. Autom.*, vol. 17, pp. 280–289, 2001.

[61] S. Kajita, F. Kanehiro, K. Kaneko, and K. Fujiwara, "Biped walking pattern generation by using preview control of zero-moment point," in *IEEE International Conference on Robotics and Automation ,ICRA 2003*, 2003, pp. 1620–1626.

[62] S. Kajita, T. Nagasaki, K. Kaneko, and H. Hirukawa, "ZMP-based biped running control," *Robot. Autom. Mag. IEEE*, vol. 14, no. 2, pp. 63–72, 2007.

[63] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3D linear inverted pendulum mode: a simple modeling for a biped walking pattern generation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001, pp. 239–246.

[64] H. Dallali, M. Brown, and B. Vanderborght, "Using the torso to compensate for non-minimum phase behaviour in zmp bipedal walking," in *Advances in Robotics Research*, Springer, 2009, pp. 191–202.

[65] S. Kagami, K. Nishivaki, M. Inaba, and H. Inoue, "A Fast Dynamically Equilibrated Walking Trajectory Generation Method of Humanoid Robot," *Auton. Robots*, vol. 12, no. 1, pp. 71–82, 2002.

[66] S. Hong, Y. Oh, D. Kim, and B.-J. You, "A walking pattern generation method with feedback and feedforward control for humanoid robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 1078–1083.

[67] J. Park and Y. Youm, "General ZMP preview control for bipedal walking," in *2007 IEEE International Conference on Robotics and Automation*, 2007, pp. 2682–2687.

[68] R. Kurazume, T. Hasegawa, and K. Yoneda, "The sway compensation trajectory for a biped robot," in *IEEE International Conference on Robotics and Automation*, 2003, pp. 925–931.

[69] A. Takanishi, H. Lim, M. Tsuda, and I. Kato, "Realization of dynamic biped walking stabilized by trunk motion on a sagittally uneven surface," in *IEEE International Workshop on Intelligent Robots and Systems Towards a New Frontier of Applications*, 1990, p. 323-330 .

[70] K. Erbatur and O. Kurt, "Natural ZMP Trajectories for Biped Robot Reference Generation," *IEEE Trans. Ind. Electron.*, vol. 56, no. 3, pp. 835–845, Mar. 2009.

[71] Y. Choi, B. J. You, and S. R. Oh, "On the stability of indirect ZMP controller for biped robot systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2004, pp. 1966–1971.

[72] K. Nishiwaki, S. Kagami, Y. Kuniyoshi, M. Inaba, and H. Inoue, "Online generation of humanoid walking motion based on a fast generation method of motion pattern that follows desired ZMP," in *IEEE/RSJ International Conference on Intelligent Robots and System*, 2002, pp. 2684–2689.

[73] I.-W. Park and J.-Y. Kim, "Fourier series-based walking pattern generation for a biped humanoid robot," in *IEEE-RAS International Conference on Humanoid Robots*, 2010, pp. 461–467.

[74] M. Zefran and V. Kumar, "Two methods for interpolating rigid body motions," in *IEEE International Conference on Robotics and Automation*, 1998, pp. 2922–2927.

[75] R. Ferreira, L. Reis, A. Moreira, and N. Lau, "Development of an Omnidirectional Kick for a NAO Humanoid Robot," in *Advances in Artificial Intelligence – IBERAMIA 2012, Lecture Note in Computer Science*, vol. 7637, Springer, 2012, pp. 571–580.

[76] A. D. Kuo and J. M. Donelan, "Dynamic principles of gait and their clinical implications.," *Phys. Ther.*, vol. 90, pp. 157–174, 2010.

[77] T. Carey and R. Crompton, "The metabolic costs of 'bent-hip, bent-knee'walking in humans," *J. Hum. Evol.*, 2005.

[78] S. Sakka, C. Hayot, and P. Lacouture, "A generalized 3D inverted pendulum model to represent human normal walking," in *10th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2010*, 2010, pp. 486–491.

[79] A. D. Kuo, J. M. Donelan, and A. Ruina, "Energetic consequences of walking like an inverted pendulum: step-to-step transitions.," *Exerc. Sport Sci. Rev.*, vol. 33, no. 2, pp. 88–97, 2005.

[80] J. E. Pratt and S. V. Drakunov, "Derivation and Application of a Conserved Orbital Energy for the Inverted Pendulum Bipedal Walking Model," in *In IEEE International Conference on Robotics and Automation (ICRA)*, 2007, pp. 4653–4660.

[81] C. Chevallereau and Y. Aoustin, "Optimal reference trajectories for walking and running of a biped robot," *Robotica*, vol. 19, no. 05, pp. 557–569, 2001.

[82] A. J. Ijspeert, "Central pattern generators for locomotion control in animals and robots: a review.," *Neural networks*, vol. 21, no. 4, pp. 642–53, May 2008.

[83] F. Delcomyn, "Neural basis of rhythmic behaviour in animals," in *Science*, vol. 210, 1980, pp. 492–498.

[84] P. S. G. Stein, *Neurons, Networks, and Motor Behavior*. MIT Press, 1999, p. 319.

[85] J. Duysens, "Human gait as a step in evolution.," *Brain A J. Neurol.*, vol. 125, pp. 2589–2590, 2002.

[86] A. D. McClellan and W. Jang, "Mechanosensory inputs to the central pattern generators for locomotion in the lamprey spinal cord: resetting, entrainment, and computer modeling.," *J. Neurophysiol.*, vol. 70, no. 6, pp. 2442–2454, 1993.

[87] J. T. Buchanan, "Neural network simulations of coupled locomotor oscillators in the lamprey spinal cord.," *Biol. Cybern.*, vol. 66, no. 4, pp. 367–74, 1992.

[88] T. Zielińska, "Coupled oscillators utilised as gait rhythm generators of a two-legged walking machine," *Biol. Cybern.*, vol. 74, no. 3, pp. 263–273, 1996.

[89] A. C. D. P. Filho, M. S. Dutra, and L. S. C. Raptopoulos, "Modeling of a bipedal robot using mutually coupled Rayleigh oscillators.," *Biol. Cybern.*, vol. 92, no. 1, pp. 1–7, 2005.

[90] J. Acebrón, L. Bonilla, C. Pérez Vicente, F. Ritort, and R. Spigler, "The Kuramoto model: A simple paradigm for synchronization phenomena," *Rev. Mod. Phys.*, vol. 77, no. 1, pp. 137–185, 2005.

[91] L. Righetti, J. Buchli, and A. Ijspeert, "Dynamic Hebbian learning in adaptive frequency oscillators," *Phys. D Nonlinear Phenom.*, vol. 216, no. 2, pp. 269–281, Apr. 2006.

[92] G. Liu, M. Habib, K. Watanabe, and K. Izumi, "The Design of Central Pattern Generators Based on the Matsuoka Oscillator to Generate Rhythmic Human-Like Movement for Biped Robots.," *J. Adv. Comput. Intell. Intell. Informatics*, vol. 11, no. 8, pp. 946–955, 2007.

[93] S. H. Strogatz, *Nonlinear Dynamics and Chaos: with applications to physics, biology, chemistry, and engineering*. Perseus Books Publishing, 1994, p. 498.

[94] A. Pikovsky, M. Rosenblum, and J. Kurths, *Synchronization: A Universal Concept in Nonlinear Sciences (Cambridge Nonlinear Science Series)*. Cambridge University Press, 2003, p. 411.

[95] R. Kempter, W. Gerstner, and J. Van Hemmen, "Hebbian learning and spiking neurons," *Phys. Rev.*, vol. 59, no. 4, pp. 4498–4514, 1999.

[96] L. Righetti and A. J. Ijspeert, "Programmable central pattern generators: an application to biped locomotion control," in *IEEE International Conference on Robotics and Automation, ICRA 2006.*, 2006, pp. 1585–1590.

[97] Q. Wu, C. Liu, J. Zhang, and Q. Chen, "Survey of locomotion control of legged robots inspired by biological concept," *Sci. China Ser. F Inf. Sci.*, vol. 52, no. 10, pp. 1715–1729, Oct. 2009.

[98] H. F. Yu, E. H. K. Fung, and X. J. Jing, "An Improved ZMP-Based CPG Model of Bipedal Robot Walking Searched by SaDE," *ISRN Robot.*, vol. 2014, pp. 1–16, 2014.

[99] J. Van Den Kieboom, "Biped Locomotion and Stability," University of Groningen, 2009.

[100] H. Picado, M. Gestal, N. Lau, L. Reis, and A. Tomé, "Automatic generation of biped walk behavior using genetic algorithms," in *Bio-Inspired Systems: Computational and Ambient Intelligence*, 2009, pp. 805–812.

[101] A. Seekircher, J. Stoecker, S. Abeyruwan, and U. Visser, "Motion capture and contemporary optimization algorithms for robust and stable motions on simulated biped robots," in *RoboCup 2012: Robot Soccer World Cup XVI*, vol. 7500, X. Chen, P. Stone, L. E. Sucar, and T. van der Zant, Eds. Springer Berlin Heidelberg, 2013, pp. 213–224.

[102] L. Boutin, A. Eon, S. Zeghloul, and P. Lacouture, "From human motion capture to humanoid locomotion imitation Application to the robots HRP-2 and HOAP-3," *Robotica*, vol. 29, no. 02, pp. 325–334, May 2010.

[103] N. Shafii, S. Aslani, O. Nezami, and S. Shiry, "Evolution of biped walking using truncated fourier series and particle swarm optimization," in *RoboCup 2009: Robot Soccer World Cup XIII, Lecture Note in Computer Science*, vol. 5949 LNAI, 2010, pp. 344–354.

[104] N. Shafii, M. H. Javadi, and B. Kimiaghalam, "A truncated fourier series with genetic algorithm for the control of biped locomotion," in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2009, pp. 1781–1785.

[105] N. Shafii, A. Khorsandian, A. Abdolmaleki, and B. Jozi, "An optimized gait generator based on fourier series towards fast and robust biped locomotion involving arms swing," in *IEEE International Conference on Automation and Logistics*, 2009, pp. 2018–2023.

[106] H. Elftman, "The function of arms in walking," *Hum. Biol.*, vol. 11, pp. 529–535, 1939.

[107] A. Pépin, K. E. Norman, and H. Barbeau, "Treadmill walking in incomplete spinal-cord-injured subjects: 1. Adaptation to changes in speed," *Spinal cord Off. J. Int. Med. Soc. Paraplegia*, vol. 41, no. 5, pp. 257–270, 2003.

[108] R. N. Hinrichs, "Whole Body Movement: Coordination of Arms and Legs in Walking and Running," *Mult. Muscle Syst. Biomech. Mov. Organ.*, pp. 534–541, 1990.

[109] E. P. Zehr and J. Duysens, "Regulation of arm and leg movement during human locomotion," *Neurosci. a Rev. J. bringing Neurobiol. Neurol. psychiatry*, vol. 10, no. 4, pp. 347–361, 2004.

[110] Y. Li, W. Wang, R. H. Crompton, and M. M. Gunther, "Free vertical moments and transverse forces in human walking and their role in relation to arm-swing," *J. Exp. Biol.*, vol. 204, pp. 47–58, 2001.

[111] K. Osaku, H. Minakata, and S. Tadakuma, "A study of CPG based walking utilizing swing of arms," in *9th IEEE International Workshop on Advanced Motion Control*, 2006, pp. 392–396.

[112] M. P. Murray, S. B. Sepic, and E. J. Barnard, "Patterns of sagittal rotation of the upper limbs in walking," *Phys. Ther.*, vol. 47, no. 4, pp. 272–284, 1967.

[113] J. Kulk and J. S. Welsh, "Evaluation of walk optimisation techniques for the NAO robot," in *2011 11th IEEE-RAS International Conference on Humanoid Robots*, 2011, pp. 306–311.

[114] B. Pimentel, L. Nuno, and L. P. Reis, "Flexible Movement Planning in Humanoid Soccer," in *Proceedings of the 10th Conference on Mobile Robots and Competitions (Robotica)*, 2010, pp. 91–96.

[115] L. Cruz, L. P. Reis, N. Lau, and A. Sousa, "Optimization Approach for the Development of Humanoid Robots' Behaviors," in *Advances in Artificial Intelligence – IBERAMIA 2012, Lecture Notes in Computer Science*, vol. 7637, Springer, 2012, pp. 491–500.

[116] S. Asta and S. Sariel-Talay, "Nature-inspired optimization for biped robot locomotion and gait planning," *Appl. Evol. Comput.*, pp. 434–443, 2011.

[117] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach, 3rd edition*. Prantice Hall, 2009.

[118] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science (80-.).*, vol. 220, no. 4598, pp. 671–680, 1983.

[119] F. Glover, "Future paths for integer programming and links to artificial intelligence," *Comput. Oper. Res.*, vol. 13, no. 5, pp. 533–549, 1986.

[120] S. Chemova and M. Veloso, "An evolutionary approach to gait learning for four-legged robots," *IEEE-RSJ Int. Conf. Intell. Robot. Syst.*, pp. 2562–2567, 2004.

[121] F. M. Silva and J. a. T. Machado, "Goal-oriented biped walking based on force interaction control," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2001, pp. 4122–4127.

[122] D. Gong, J. Yan, and G. Zuo, "A Review of Gait Optimization Based on Evolutionary Computation," *Appl. Comput. Intell. Soft Comput.*, pp. 1–12, 2010.

[123] K. Wolff, J. Pettersson, A. Heralic, and M. Wahde, "Structural evolution of central pattern generators for bipedal walking in 3D simulation," in *Proceedings of the 2006 IEEE International Conference on Systems Man and Cybernetics SMC06*, 2006, pp. 227–234.

[124] F. S. Manuel, R. S. Barbosa, and J. A. T. Machado, "Development of a genetic algorithm for the optimization of hexapod robot parameters," in *Proceedings of the International Conference Applied Simulation and Modelling (ASM 2009)*, 2009, pp. 77–82.

[125] P. Vundavilli and D. Pratihar, "Soft computing-based gait planners for a dynamically balanced biped robot negotiating sloping surfaces," *Appl. Soft Comput.*, vol. 9, no. 1, pp. 191–208, 2009.

[126] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989, p. 432.

[127] F. Herrera, M. Lozano, and J. L. Verdegay, "Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis," *Artif. Intell. Rev.*, vol. 12, no. 4, pp. 265–319, 1998.

[128] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Trans. Syst. Man Cybern.*, vol. 24, no. 4, pp. 656–667, 1994.

[129] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN95 International Conference on Neural Networks*, 1995, vol. 4, no. 3, pp. 1942–1948.

[130] C. Rong, Q. Wang, Y. Huang, G. Xie, and L. Wang, "Autonomous Evolution of High-Speed Quadruped Gaits Using Particle Swarm Optimization," in *RoboCup 2008: Robot Soccer World Cup XII, Lecture Note in Computer Science*, vol. 5399, Springer, 2009, pp. 259–270.

[131] C. Niehaus, T. Röfer, and T. Laue, "Gait optimization on a humanoid robot using particle swarm optimization," in *Soccer Humanoid workshop Proceedings of the Second Workshop on Humanoid Soccer Robots in cunjunction with IEEE-RAS International Conference on Humanoid Robotics,* 2007, pp. 1–7.

[132] B. Jiao, Z. Lian, and X. Gu, "A dynamic inertia weight particle swarm optimization algorithm," *Chaos, Solitons and Fractals*, vol. 37, pp. 698–705, 2008.

[133] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies.," *Evol. Comput.*, vol. 9, pp. 159–195, 2001.

[134] P. Macalpine and P. Stone, "Using dynamic rewards to learn a fully holonomic bipedal walk," in *AAMAS Adaptive Learning Agents (ALA) Workshop*, 2012.

[135] D. Urieli, P. Macalpine, S. Kalyanakrishnan, Y. Bentor, and P. Stone, "On Optimizing Interdependent Skills : A Case Study in Simulated 3D Humanoid Robot Soccer Categories and Subject Descriptors," in *10th International Conference on Autonomous Agents and Multiagent Systems*, 2011, pp. 769–776.

[136] J. E. A. Bertram, "Constrained optimization in human walking: cost minimization and gait plasticity," *J. Exp. Biol.*, vol. 208, no. 6, pp. 979–991, 2005.

[137] J. C. Zagal and J. Ruiz-del-Solar, "Combining Simulation and Reality in Evolutionary Robotics," *J. Intell. Robot. Syst.*, vol. 50, no. 1, pp. 19–39, 2007.

[138] T. Laue and M. Hebbel, "Automatic Parameter Optimization for a Dynamic Robot Simulation," in *RoboCup 2008 Robot Soccer World Cup XII, Lecture Note in Computer Science*, vol. 5399, Springer, 2009, pp. 121–132.

[139] S. Kalyanakrishnan, T. Hester, M. Quinlan, Y. Bentor, and P. Stone, "Three Humanoid Soccer Platforms : Comparison and Synthesis," in *RoboCup 2009 Robot Soccer World Cup XIII, Lecture Note in Computer Science*, vol. 5949, Springer, 2010, pp. 140–152.

[140] P. Costa, J. Lima, J. Gonçalves and P. Malheiros, "SimTwo Realistic Simulator: A Tool for the Development and Validation of Robot Software," *Appl. Math.*, vol. 1, pp. 17–33, 2011.

[141] O. Obst and M. Rollmann, "Spark - A Generic Simulator for Physical Multi-Agent Simulations," in *Multiagent System Technologies, Lecture Note in Computer Science*, vol. 3187, Springer, 2004, pp. 243–257.

[142] N. Shafii, L. Reis, and R. Rossetti, "Two humanoid simulators: Comparison and synthesis," in *Proceeding of the 6th Iberian Conference onInformation Systems and Technologies (CISTI)*, 2011, pp. 1–6.

[143] F. Kanehiro, K. Fujiwara, S. Kajita, K. Yokoi, K. Kaneko, H. Hirukawa, Y. Nakamura, and K. Yamane, "Open architecture humanoid robotics platform," in *Proceedings 2002 IEEE International Conference on Robotics and Automation*, 2002, vol. 1, pp. 24–30.

[144] C. Ltd., "WebotsTM: Professional mobile robot simulation," *Int. J. Adv. Robot. Syst.*, vol. 1, no. 1, pp. 39–42, 2004.

[145] Y. Xu, "From Simulation to Reality: Migration of Humanoid Robot Control," Humboldt-Universität zu Berlin, 2014.

[146] N. Koenig and A. Howard, "Gazebo - 3d multiple robot simulator with dynamics." [Online]. Available: http://playerstage.sourceforge.net/gazebo/gazebo.html. [Accessed: 15-Jan-2015].

[147] B. P. Gerkey, R. T. Vaughan, and A. Howard, "The Player / Stage Project : Tools for Multi-Robot and Distributed Sensor Systems," in *Proceedings of the International Conference on Advanced Robotics (ICAR 2003)*, 2003, pp. 317–323.

[148] T. Rofer, "Evolutionary Gait-Optimization Using a Fitness Function Based on Proprioception," in *RoboCup 2004: Robot Soccer World Cup VIII, Lecture Note in Computer Science*, vol. 3276, Springer, 2005, pp. 310–322.

[149] L. Yang, C. M. Chew, A. N. Poo, and T. Zielinska, "Adjustable bipedal gait generation using genetic algorithm optimized fourier series formulation," in *IEEE International Conference on Intelligent Robots and Systems*, 2006, pp. 4435–4440.

[150] S. Kagami, M. Mochimaru, Y. Ehara, N. Miyata, K. Nishiwaki, T. Kanade, and H. Inoue, "Measurement and comparison of humanoid H7 walking with human being," *Rob. Auton. Syst.*, vol. 48, no. 4, pp. 177–187, 2004.

[151] D. Thewlis, J. Richards, and S. J. Hobbs, "the Appropriateness of Methods Used To Calculate Joint Kinematics," *J. Biomech.*, vol. 41, pp. 320–334, 2008.

[152] R. J. Konz, S. Fatone, R. L. Stine, A. Ganju, S. A. Gard, and S. L. Ondra, "A kinematic model to assess spinal motion during walking.," *Spine (Phila. Pa. 1976).*, vol. 31, no. 24, pp. 898–906, 2006.

[153] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Yokoi, and H. Hirukawa, "A realtime pattern generator for biped walking," in *Proceedings IEEE International Conference on Robotics and Automation*, 2002, pp. 31–37.

[154] Z. Tang, Z. Sun, H. Liu, and M. J. Er, "Clock-turning gait synthesis for humanoid robots," *J. Control Theory Appl.*, vol. 5, no. 1, pp. 23–27, Feb. 2007.

[155] M. H. Woollacott and P. F. Tang, "Balance control during walking in the older adult: research and its implications," *Phys. Ther.*, vol. 77, no. 6, pp. 646–660, 1997.

[156] H. B. Menz, S. R. Lord, and R. C. Fitzpatrick, "Acceleration patterns of the head and pelvis when walking on level and irregular surfaces.," *Gait Posture*, vol. 18, no. 1, pp. 35–46, 2003.

[157] H.-G. Beyer, "Evolutionary algorithms in noisy environments: theoretical issues and guidelines for practice," *Comput. Methods Appl. Mech. Eng.*, vol. 186, pp. 239–267, 2000.

[158] S. Behnke, "Online trajectory generation for omnidirectional biped walking," in *IEEE International Conference on Robotics and Automation, ICRA 06,* 2006, pp. 1597–1603.

[159] K. Erbatur and U. Seven, "An inverted pendulum based approach to biped trajectory generation with swing leg dynamics," in *IEEE-RAS International Conference on Humanoid Robots*, 2007, pp. 216–221.

[160] M. Yilmaz, U. Seven, K. C. Fidan, T. Akbas, and K. Erbatur, "Circular arc-shaped walking trajectory generation for bipedal humanoid robots," in *12th IEEE International Workshop onAdvanced Motion Control (AMC)*, 2012, pp. 1–8.

[161] T. Katayama, T. Ohki, T. Inoue, and T. Kato, "Design of an optimal controller for a discrete-time system subject to previewable demand," *Int. J. Control*, vol. 41, no. 3, pp. 677–699, 1985.

[162] K. Yokoi, K. Kaneko, K. T. Aist, T. Central, and T. Ibaraki, "Running Pattern Generation for a Humanoid Robot," in *IEEE International Conference on Robotics and Automation*, 2002, pp. 2755–2761.

[163] R. Tajima, D. Honda, and K. Suga, "Fast running experiments involving a humanoid robot," in *IEEE International Conference on Robotics and Automation, ICRA'09*, 2009, pp. 1571–1576.

[164] K. E. Gordon, D. P. Ferris, and A. D. Kuo, "Metabolic and Mechanical Energy Costs of Reducing Vertical Center of Mass Movement During Gait," *Arch. Phys. Med. Rehabil.*, vol. 90, pp. 136–144, 2009.

[165] P. Kormushev, B. Ugurlu, S. Calinon, N. G. Tsagarakis, and D. G. Caldwell, "Bipedal walking energy minimization by reinforcement learning with evolving policy parameterization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 318–324.

[166] P. MacAlpine, S. Barrett, D. Urieli, V. Vu, and P. Stone, "Design and optimization of an omnidirectional humanoid walk: A winning approach at the RoboCup 2011 3D simulation competition," in *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI-12)*, 2012, pp. 1047–1053.

[167] A. Farchy, S. Barrett, P. MacAlpine, and P. Stone, "Humanoid robots learning to walk faster: From the real world to simulation and back," in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, 2013, pp. 39–46.